



Esprit '91

Conference Proceedings

Commission of the European Communities
DG XIII: Telecommunications, Information Industries and Innovation

Esprit '91

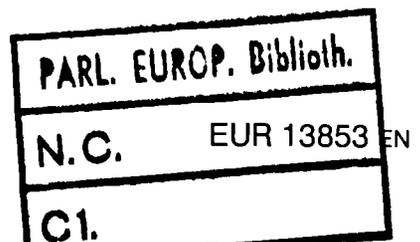
Esprit '91

Proceedings of the Annual Esprit Conference,
Brussels, 25-29 November 1991

Edited by

Commission of the European Communities
Directorate-General
Telecommunications, Information Industries and Innovation

1991



Published by the
COMMISSION OF THE EUROPEAN COMMUNITIES
Directorate-General
Telecommunications, Information Industries and Innovation
L-2920 Luxembourg

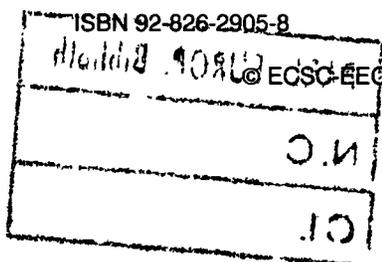
LEGAL NOTICE

Neither the Commission of the European Communities nor any person acting on behalf of the Commission is responsible for the use which might be made of the following information

Cataloguing data can be found at the end of this publication

Luxembourg: Office for Official Publications of the European Communities, 1991

Catalogue number: CD-NA-13853-EN-C



Printed in Belgium

FOREWORD

The 1991 ESPRIT Conference is being held in Brussels from 25-29 November. Well over 2000 participants from all over Europe and beyond are expected to attend. The Conference offers a unique opportunity to be updated on the results of the ESPRIT projects and Basic Research actions and to develop international contacts with colleagues working in every area of Information Technology (IT).

The first three days of the Conference are devoted to presentations of ESPRIT projects and Basic Research actions in plenary and parallel sessions. The scope of the Conference has again been broadened this year by the inclusion of several well-known international speakers. All areas of ESPRIT work are covered: Microelectronics, Information Processing Systems and Software, Advanced Business and Home Systems Peripherals, Computer Integrated Manufacturing and Engineering, the new Open Microprocessor Systems Initiative, Basic Research, and the Information Exchange System.

During the IT Forum on 28 November, prominent European industrial leaders including manufacturers as well as users of IT will address issues of major importance for this industry.

More than 140 projects and actions will display their major innovations and achievements at the expanded ESPRIT Exhibition, which will again be open to the general public. The exhibition offers an excellent opportunity to acquire comprehensive knowledge about the results obtained in all areas of the ESPRIT programme. These exhibits also show the pervasive role of IT and the contribution it makes to the economic and social life of everyone in the Community.

I would like to congratulate and thank everyone who has contributed to the Conference: the Programme Committee in charge of selecting papers; the authors and reviewers of the papers themselves; the chairmen, speakers, panellists and workshop participants at the Conference; the project teams which set up the Exhibition demonstrations. The success of this Conference is due to all their efforts.

J-M. Cadiou
Director
ESPRIT

ACKNOWLEDGEMENTS

The Commission of the European Communities thanks the following members of the Conference Programme Committee for their contribution to the 1991 ESPRIT Conference.

G. Aulin
F. Bancelhon
J. Campbell
L. Claesen
F. Craven
J. Goethals
R. Killick
M. Reeve
N. Rischette
D. Slight
A. Stoneham
E. Trostmann
I. Watson

CONTENTS

Foreword	v
Acknowledgements	vi
Table of Contents	vii

Plenary Session

SCOPE, Achievements and Perspectives (<i>Project 2151</i>) Robert Ph.	3
The SPIRIT High Performance Technical Workstation (<i>Project 2484</i>) Wieczorek R., Grimsdale R., Leclerc P., de Lange M., Strasser W., Beal D.	18
The Esprit Effect on CIM Development from Strategic to Operative Level: The Pirelli Tyre Holding Experience (<i>Project 2434</i>) Basaglia G., Guida M.	34
Open Microprocessor Systems Initiative (<i>Project 5386</i>)	54
Nanoelectronics: A Vision of the Future? (<i>Action 3133</i>) Beaumont S.	80

Microelectronics

TIPBASE - Bipolar Advanced Silicon for Europe (<i>Project 2016</i>) Pruijmboom, A.	91
Achievements in High Density Packaging (<i>Project 2075</i>) Chantraine P., Chandler N., Brandenburger J., Chilo J., de Maquillé Y., O'Mathuna C., Koschnick W., Bargain R., Dümcke R.	101
EDIF as a Standard Test Specification Format (<i>Project 2318</i>) Vandelloo P., Verhelst B., Wahl M.	112
N/A Research Into Boundary Scan Test Implementation (<i>Project 2478</i>) Kritter S., Rahaga T.	121
Planet MOVPE Equipment: a Breakthrough in Industrial Production of Sophisticated Epitaxial Layers (<i>Project 5003</i>) Hollan L., Frijlink P.M., Marchal J.M., Waucquez Ch., Juergensen H., Strauch G., Hernandez-Gil Gomez J.F., Acket G.A., Ambrosius H.P.M., Calleja Pardo E.	135
Development of a Low Stress High Pincount Plastic Package for High Reliability CMOS ASICs (<i>Project 5033</i>) Claes A., Kelly G., Exposito J., Neef G., Zachariassen K.	152
Rapid Thermal Processing Equipment Design and Integration for a Cluster Tool Module (<i>Project 5041</i>) Theiler T.	165

Common Library Concept and Design Methodology in IDPS (<i>Project 5075</i>) Moreau JP.	174
---	-----

Information Processing Software and Systems

DBS3, an Implementation of the EDS DBMS on a Shared-Memory Multiprocessor (<i>Project 2025</i>) Bergsten B., Couprie M., Valduriez P.	191
An Introduction to Distributed Programming in REX (<i>Project 2080</i>) Kramer J., Magee J., Sloman M., Dulay N., Cheung SC., Crane S., Twidle K.	207
Adverse-Environment Recognition of Speech (A.R.S.) (<i>Project 2101</i>) Babini G., Fissore L., Ainsworth W., Frangoulis E., Garcia Gomez R., Grenier Y., Lockwood Ph.	222
Delta-4 Architecture Validation (<i>Project 2252</i>) Kanoun K., Ariat J., Burrill L., Crouzet Y., Graf S., Martins E., MacInnes A., Powell D., Richier J.-L., Voiron J.	234
Cooperation in Industrial Systems (<i>Project 2256</i>) Jennings N.R.	253
An Optical Photorefractive Correlator for Robotic Applications (<i>Project 2288</i>) Rajbenbach H., Bann S., Refregier Ph., Joffre P., Huignar J.P., Buchkremer H.St., Jensen A.S., Rasmussen E., Brenner K.H., Lohman G.	264
A New Modular Course For Teaching About Software Engineering Measurement Within Academia (<i>Project 2384</i>) Bush M., Russell M.	278
Databases and Executable Temporal Logic (<i>Project 2469</i>) Finger M., McBrien P., Owens R.	288
Graphical Interaction in a Multimodal Interface (<i>Project 2474</i>) Ben Amara H., Peroche B., Chappel H., Wilson M.D.	303
The Transfer of Vision Research to Vehicle Systems and Demonstrations (<i>Project 2502</i>) Buxton B.F., Arrighetti S., Castelow D.A., Ferrari F., Harris C.G., Magrassi M., Masciangelo S., McLauchlan P., Moron P., von der Nuell S., Rygol M., Sandini G., Vaillant R., Wang H.	322
Interconnection Networks for Universal Message-Passing Systems (<i>Project 2701</i>) Klein A.	336
Using the Genesis Distributed Memory Benchmarks to Evaluate the SUPRENUM Computer (<i>Project 2702</i>) Addison C., Bishop N., Hey T., Hockney R., Wolton I.	352
CASTLE: A Tool for Bayesian Learning (<i>Project 5170</i>) Acid S., de Campos L.M., Gonzalez A., Molina R., Perez de la Blanca N.	363

Macro and Micro Features for Automated Pronunciation Improvement in the Spell System (<i>Project 5192</i>) Hiller S.M., Rooney E., Laver J., Di Benedetto M.-G., Lefèvre J.-P.	378
Neurocomputing (<i>Project 5293</i>) Angéniol B., Fogelman F., Marcadé E., Pimont J.-M., Giry Ph., Simon C., Ramacher U., Theeten J.B., Friedel P., Makram S., Treleaven P., Recce M.	393
Advanced Business and Home Systems - Peripherals	
SPRITE A System for Technical Documentation (<i>Project 2001</i>) Hoppe J.	409
Security Levels Supported by the COMANDOS Security Architecture (<i>Project 2071</i>) Medina M., Moreno A.	421
Supporting Object Oriented Languages on the Comandos Platform (<i>Project 2071</i>) Cahill V., Horn C., Starovic G., Lea R., Sousa P.	427
High Fidelity, Programmable CCD Colour Camera for Desk Top Publishing and the Graphic Arts (<i>Project 2103</i>) Brunner H., Engelhardt K., Gale M.T., Lang G.K., Metzler P., Raynor J.M., Seitz P., Brissot L., Cilia G., Gadda C., Leone V.	439
Fault Tolerant Data Management in PORDOS (<i>Project 2146</i>) Becker T.	455
The Translator's Workbench - An Environment for Multi-Lingual Text Processing and Translation (<i>Project 2315</i>) Kugler M., Höge M., Heyer G., Kese R., v. Kleist-Retzow B., Winkelmann G.	470
On Mobile Work and Elusive Offices: The Elusive Office Project (<i>Project 2382</i>) Korte W.B., Robinson S., Kordey N., Keil K., Heuschötter P., Nachtsheim R.	479
Resource Management in the Home System (<i>Project 2431</i>) Fanshawe D.G.J.	506
The ARGOSI Project for ISO/IEC Graphics and Networking Standards (<i>Project 2463</i>) Bardyn J.-J., Day R.A., Duce D.A., Gallop J.R., Mistral L., Sutcliffe D.C.	512
An Advanced Coprocessor Architecture for Fast Public Key Cryptography at Variable Key Length (<i>Project 2704</i>) Schützeneder H., Eberhard G.	529
Objects + Scripts = Applications (<i>Project 2705</i>) Nierstrasz O., Tschritzis D., de Mey V., Stadelmann M.	534
Establishing the Conditions for Success in Multimedia CD Publishing (<i>Project 5656</i>) Bulthuis W.	553

Computer Integrated Manufacturing and Engineering

Active Vibration Control of Industrial Robots - Final Results of ESPRIT Project SACODY (<i>Project 1561</i>) Faillot J.-L.	563
Neutral Product Definition Database for Large Multifunctional Systems - Neutrabas (<i>Project 2010</i>) Fernández-Gonzalez F., Lehne MG., Vopel R.	578
EPIC - An Integrated Environment for Preliminary Process and Control Design (<i>Project 2090</i>) Larintzakis M., Papafotiou K., Mantzari P., Efthimiadis A.	593
Holographic Labelling in CIM Environment (<i>Project 2127</i>) Hauck R., Halewood S.J., Bernardo L.M., Powell K., Mazzocchi G.	601
ISDN Mobile Terminals via Indoor Diffuse Infrared Channel (<i>Project 2198</i>) Roviras D., Lescure M., Chapuis C., Meschenmoser R.	615
Fatigue Analysis and Acoustic Radiation Prediction in View of Better Car Design (<i>Project 2486</i>) Leuridan J., Vandeurzen U., van der Auweraer H.	628
Driving Robots via Neutral Interfaces (<i>Project 2614</i>) Kroszynski U.I., Sørensen T., Clausen T.C., Trostmann E.	646
Methods for Advanced Group Technology Integrated with CAD/CAM (<i>Project 2623</i>) de Moor D.	661

Information Exchange System

The COSINE Project (<i>Project 710</i>) Davies D.R.H., Goodman D., Knight G., Romaguera J.	675
Y-Net the ESPRIT Pan-European Community OSI Network Berrino C., Manuello D.	692

Basic Research

Building User Interfaces: Organizing Software Agents (<i>Action 3066</i>) Nigay L., Coutaz J.	707
Design Space Analysis: Bridging from Theory to Practice via Design Rationale (<i>Action 3066</i>) Maclean A., Young R., Bellotti V., Moran T.	720
Maps as Bulk Types for Data Base Programming Languages (<i>Action 3070</i>) Atkinson M., Lécluse C., Philbrow P., Richard Ph.	731

Towards a Unification of Rewrite Based Optimization Techniques for Object-Oriented Queries (<i>Action 3070</i>) Cluet S., Delobel C.	758
Observation of Photoluminescence form InAs Surface Quantum Wells Grown on InP (100) by Molecular Beam Epitaxy (<i>Action 3086</i>) Sobiesierski Z., Clark S.A., Williams R.H., Tabata A., Benyattou T., Guillot G., Gendry M., Hollinger G., Viktorovitch P.	773
InAs/InP Strained Quantum Wells Grown by Hydride Vapor Phase Epitaxy and Studied by Photoluminescence (<i>Action 3086</i>) Leymarie J., Mihailovic M., Gil E., Piffault N., Vasson A., Vasson A.M., Tabata A., Benyattou T., Guillot G., Cadoret R.	780
Fault Injection for the Experimental Validation of Fault Tolerance (<i>Action 3092</i>) Arlat J., Crouzet Y., Laprie J.-Cl.	791
Reliability Analyses of Workstation Failure Data (<i>Action 3092</i>) Brocklehurst S., Kanoun K., Laprie J.C., Littlewood B., Metge S., Mellor P., Tanner A.	806
Towards the MEDLAR Framework (<i>Action 3125</i>) Cunningham J., Gabbay D., Ohlbach H.J.	822
FOF Production Theory: Towards an Integrated Theory for One-of-a-Kind Production (<i>Action 3143</i>) Wortmann J.C.	842
Turning the Formal Verification of VLSI Hardware into Reality (<i>Action 3216</i>) Claesen L., Borrionel D., Evekings H., Paillet J.L., Prinetto P.	857

Indexes

Index of authors	875
Index of Project Numbers	879
Index of Acronyms	881

PLENARY SESSION

SCOPE, Achievements and Perspectives

Philippe Robert
VERILOG
150 Rue N. Vauquelin
F 31081 Toulouse Cedex
Tel : (+ 33) 61 19 29 39
e-mail : Philippe_Robert_VERILOG@eurokom.ie

Abstract

The SCOPE project is, within the ESPRIT framework, aimed at the development and promotion of software product assessment for certification, which is highly needed to strengthen and integrate the European software industry for the 1993 open market. A common yardstick is necessary to improve the quality of software products by clarifying the relations between suppliers and customers, to offer a means of competition by allowing third party certification, and to promote the usage of best engineering practices by demonstrating their impact on quality.

The issues that are dealt with by SCOPE include the possibility of obtaining the consensus needed for standardization and the economic aspects of software evaluation. The results presented here focus on an assessment methodology that includes proposals for standardized measurement procedures. This methodology and these procedures are being experimented on industrial case studies, providing evidence of their applicability concerning technical and economic aspects. In addition to the technical aspects, the legal issues related to software and certification are being investigated. The perspectives of implementation of the project results, as well as their expected impact on industry are finally discussed.

Introduction

The ESPRIT Project 2151 SCOPE started more than two years ago (in March 1989). SCOPE is an acronym for Software CertificatiOn Programme in Europe. The goal of this project is to develop an efficient methodology and associated technology for software product assessment and to demonstrate their applicability.

Software products assessment is highly important to set up a truly integrated European software industry for the 1993 open market. Our approach is to build on available experience and on results achieved by R&D during the past ten years; emphasis is put on experimentation on industrial case studies rather than development of new techniques. The consensus elaborated within SCOPE is to be widened via standardization and information dissemination.

The aim of this paper is to present SCOPE's first results and to describe their implementation perspectives in the software industry.

Organization of the paper

The first part of this paper presents the objectives of the project in the context of European Software industry needs and of other certification initiatives. We then focus on the major achievements of the first part of the project so far: definition of a general method for assessing software products, as well as standardized assessment proce-

dures, investigation of the legal aspects of certifying software products and practical experience of software assessment. Finally, important lines of action for the rest of the project are described. They will lead to a set of results that will be of benefit to the European software industry.

More general information on the project can be found in [1].

1. Objectives and Background

1.1. Objectives of the project

Certification in the software industry is an issue that has been generating a lot of different initiatives in the past few years. It is therefore necessary to locate precisely our action in that field. In this context, we study the technical assessment of software products for their certification. Certification is an administrative process, executed by an independent, third party organization (certification agency), that may, after review of an assessment report, deliver a certificate or label to a product. The assessment is conducted by a test laboratory which produces an assessment report which can later be used by the certification agency.

This point has much influence on our work since fairness and reproducible results are required for a European certification scheme that will be using the assessment technology developed by SCOPE. Such a scheme could be implemented in the framework of the newly created EOTC (European Organization of Testing and Certification) that promotes the setting-up of "recognition arrangements" between European test laboratories, producers, users and certification agencies.

From a technical point of view, our approach is aimed at being general. Therefore, we consider software products in a broad sense: by software product we mean documents that can be delivered to a user. This may vary with the case; it will include, at least, executable code and user documentation. When appropriate, development documents such as source code, design and requirements specification or traces of the development process such as test cases or review reports, will be part of the product.

Those documents are measured and inspected to assess the conformity of the whole product with regard to a specification which describes functional and non-functional requirements. The requirements may be extracted from the actual requirement specification that was used as a basis for the product development. They are elaborated by the evaluation "sponsor" who can be the software producer, a specific user or any other organization.

Although the technology developed by SCOPE is constrained by the requirements for third party evaluation, we consider that its use in industry will extend beyond this context and that standardized assessment techniques are needed outside certification.

1.2. Why Assess Software Products?

SCOPE is a large project involving 12 partners and 4 associate partners. The effort planned is around 100 person.year. The justification of such an investment is the need for Europe to improve the strength of its software industry by promoting best practices and favouring competition.

The context

The first aspect to be considered is the growing importance of software in all domains of the economy. No matter if software is used in company management, in production process or directly included in company products, the costs of its non-conformity to requirements is raising more and more concern. Moreover, software is often acknowledged as being of poor quality.

Parallel to economic aspects, the development of European directives aiming at the harmonization of technical regulations is a potential source of software assessment needs. Conformity to directives for the ever growing number of products including software components might imply, to be demonstrated, the assessment of embedded software. This is especially true in the case of safety or security critical systems.

In this context, it is of paramount importance to provide the actors of the European software industry with common means of measurement and assessment in order to reduce the losses due to bad quality software and, therefore, to improve the practices for a better competitiveness of the European industry.

The needs

The costs of bad quality software can be reduced by clarifying the relationship between suppliers and users of software. This requires largely accepted methods to state and assess quality requirements that could be used in contractual documents.

Such methods will be the basis for third party testing and certification of software products that would improve competition between software producers by helping customers to choose between alternative products. This is needed to increase the level of business between European companies in the field of software.

Furthermore, software product evaluation can be seen as a means of promoting the use of modern Software Engineering techniques in industry. The development of a methodology for software product assessment leads to recognize the best practices in Software Engineering; they can therefore be more easily accepted by industry. Any such certification scheme will, when being set up, involve a larger cooperation between industrial companies that will reinforce the European leadership in Software Engineering. Finally, certification may be used to justify the introduction of new techniques and methods within companies.

All these aspects together make the assessment and certification of software products a major issue for the future of Europe. In this context, SCOPE is a key project in the ESPRIT programme that is working to propose common means of software assessment needed to build a strong, European wide, software industry.

1.3. Background of the project

A number of activities exist that are related to certification in the software industry. It is interesting to relate some of them to SCOPE in order to understand its aims more clearly. The list of activities presented below is in no way complete or detailed and more can be found in the SCOPE public reports R 1.1.1.1 "Software Certification: State of the Art" [2] and R 2.6.3.1.1 "Software Certification: The European Context" [3].

Certification in the software industry has often been identified, in the past few years, to the auditing of Quality Assurance Systems. This approach, also very different from ours, is an important source of experience and is taken into account in SCOPE. In the domain of software for safety critical systems, the practice of certification is, as well, a

useful background. Another domain is that of software for which "functional" standards exist and that can be "validated"; compilers are good examples. Finally, the *Gütegemeinschaft Software*, in Germany has set up a certification scheme for software packages that is extremely relevant to us.

1.3.1. Certification of Quality Systems

One way of ensuring that software meets its requirements is to implement and apply a good quality management system for its production. Assessing an organization quality system should give confidence in the quality of the products that come out of that organization. A third-party assessment of software quality systems, based on the EN 29000 series of standards, has been developed under the Conformance Testing Services programme (CTS) of the CEC [4].

Key issues for assessment are quality policy and organization, design method, tests and reviews and the configuration management system.

A successful third-party assessment results in the delivery to the organization of a label which can be used to promote its activities. After that, the producer must undergo regular third-party controls in order to keep its certificate.

This type of certification is definitely complementary to our approach: product assessment will increase the confidence in individual products, while a certified quality system might simplify the assessment of successive versions of a product.

1.3.2. High Criticality Software

In civil aviation, railways and nuclear industry, software certification is a result of statutory requirements for systems safety (aircraft, signalling, power plants); the recent introduction of software in critical parts of these systems has led to the development of certification schemes for software; the example of civil aviation is presented below.

Airworthiness certification has been mandatory for aircrafts for a long time now. The implementation of safety critical systems using computers in recent aircraft programmes has led the civil aviation authorities to develop a specific standard for software (RTCA Do 178-A).

In this context, the method that leads to certification is a good example of the "process certification" approach:

- certifier and manufacturer agree on the criticality of the software depending on the potential consequences of a software failure;
- they then agree on a software development methodology and certification plan, since no single development methodology is recommended by Do 178-A;
- the software is developed, tested and validated according to the methodology; each of the corresponding activities must produce documentary evidence of their application and exhibit traceability;
- this evidence is analyzed by the certifier who grants the certificate.

The limitation of this approach may be that final product quality is not proven and, for larger scale application, the cost of producing certifiable products. This experience provides us with conditions for applicability of process assessment (for us, process traces are potential product parts). Alternatively, we hope that some of the assessment procedures we develop may be used in the context of airworthiness certification.

1.3.3. Validation of Compilers

A certification scheme exists for several languages for which there is a standard definition (for example, Fortran or Ada). This standard definition specifies the functionalities expected from a compiler with respect to syntactic and semantic aspects, error detection and generated code.

The certification scheme relies on the design of test suites which are submitted to the compiler under validation, resulting in compiling errors with diagnostics or production of executable code. This method is thus purely product-oriented. Several of these certification schemes have been set up in Europe under the Conformance Testing Services (CTS) programme.

The advantages of this approach to software certification are its relatively limited cost and its good acceptance and understanding by producers and customers alike.

However, this approach has two major limitations. Firstly, the correctness of the compiler might be limited to the cases that appear in the test suite. No other quality attributes, such as performance, are assessed. The second limitation is the lack of generality; to set up such a scheme, there is a need for an a-priori standard definition of the certifiable characteristics (mainly functionalities) of the product. Several types of product, such as communication components or graphical packages have a similar validation scheme. However, this approach cannot be extended to the majority of software products, due to their diversity and to the cost of developing specific test suites.

1.3.4. The Gütegemeinschaft Software

In Germany, software quality labels are awarded by Gütegemeinschaft Software (GGS), which is mainly dedicated to software packages without safety relevance. The approach used is purely product-oriented and the label is granted on the basis of:

- the software product description (which must clearly define what the product should and should not do),
- the software product itself (limited to the executable code and its documentation), which must satisfy the description.

The test procedure is defined by a standard [5] giving the requirements for product description and specifying the use of "black-box testing" techniques to ensure the consistency of the product with its description.

The GGS scheme is interesting for SCOPE, since it addresses a wide range of software and uses the "product" approach. However, the fact that only correctness and documentation quality are assessed and that embedded software is not addressed is a limitation.

1.4. SCOPE Philosophy

One of the premises of the project is that the Technical State of the Art of Software Engineering is ready for software certification. The Software Engineering community has been active for a long time and there are claims that there is enough R&D and experimentation in metrics and assessment techniques to identify the kernel which can be applied to industrial projects for quality evaluation. A part of this knowledge and some techniques are now ready to be transferred to industry for implementation of

Software Certification. The success of this transfer will be the source of new experience and data which will surely open new perspectives for research.

The approach chosen by SCOPE is to use "state of the art" measurement techniques to combine them in order to assess software products. The techniques which are taken into consideration concern elements of a software product, such as executable code, user documentation of source code, and traces of the development process, such as test cases, review reports or quality plans.

Another important aspect of the SCOPE philosophy is to assess the conformity of the product to stated requirements. This means that no assessment will be the same as another one since there are no identical software product to each other. Moreover, these requirements not only concern functionalities but also quality characteristics such as efficiency, reliability or maintainability. Depending of the case, the emphasis on these characteristics may vary.

In the SCOPE project, the key activities are case studies. They are based on industrial software development projects, the results of which are monitored by SCOPE partners. The goals of case studies in SCOPE are twofold. Partners describe explicitly their experience concerning software assessment and, by gathering it, define the SCOPE approach. The goal is also to demonstrate the applicability of the various aspects of the method developed.

To conclude this overview of the general SCOPE approach, it is worth stating that we cover more than the only technical aspects. A significant effort is made to investigate the legal aspects of software product certification. This is essential to contribute to the applicability demonstration of our approach and to prepare potential implementation.

1.5. Organization of the Project

To describe in detail the organization of the project is not the aim of this paper. However, some aspects may help for a better understanding of the project.

First of all, the work-plan has been divided into two sequential work-packages. The first one, now finished, was aiming at defining the approach and preparing the experimentation while the second one is the main experimentation phase with the implementation of the results as a second objective.

Another important point to discuss is the partnership of the project. The number of partners is rather large compared to an average ESPRIT project. This comes from the necessity to cover the various components of the software engineering community (academy, research, test laboratories and industry) and to run a large number of independent case studies.

The SCOPE consortium gather 12 partners and 4 associate partners:

- from academy, Dublin City University (IRL), Glasgow College (UK), Strathclyde University (UK) and The City University London (UK), Institut Catala de Tecnologia (SP);
- from research, Gesellschaft für Mathematik und Datenverarbeitung (D), VTT (SF);
- test laboratories, AEA Technology (UK), CEA (F, associate partner), ElektronikCentralen (DK), Gesellschaft für Reaktor Sicherheit (D) and TUEV Bayern (D, associate partner);
- from industry, Etnoteam (I), Veridatas (F, associate partner) and Verilog (F, coordinator);
- the legal study is conducted by Strathclyde University, Law School (UK) in collaboration with Cabinet Alain Bensoussan (F, associate partner).

2. Principle Achievements

The goal of this chapter is not to describe all the achievements of the project so far, but to give an overview of those that are considered as most significant for the rest of the project and for their impact on the software engineering community. We therefore present first the general framework for software assessment, the experimentation of which as been started in the main phase of the project. We then discuss the experience gathered concerning practical issues related with software assessment. Finally, the first findings of the SCOPE legal study are described in the context of their influence on technical work.

2.1. SCOPE Assessment Framework

The development of the SCOPE assessment methodology [6] has been the kernel technical activity during the definition phase of the project. This methodology is now under large scale experimentation. The goal has been to propose an integration framework for existing and accepted measurement and assessment that would allow the setting-up of a certification scheme for software products. This imposes constraints on the framework that include some aspects of its structure. These constraints are first presented below. We then discuss the various elements of the proposed methodology.

2.1.1. Constraints on the Framework

SCOPE's primary objective is to make it possible to set up a certification scheme for software products. It is clear, however, that the needs for such a certification scheme are not unique and that many implementations are to be expected. They will cover the needs, for example, of specific users organizations or those of a general purpose regional labelling scheme. Among the foreseeable diversity of the certification schemes that could be set up, some generic constraints on the technical issues can be identified and are discussed below: the need for fairness and reproducible results and the need for identifying the specific conformity requirements for each certification case. This section is concluded by a summary of what would constitute the specificity of a certification scheme based on SCOPE results.

2.1.1.1. The need for fairness and reproducible results

In the current state of the project, no extensive study has been yet performed on the actual strategy to implement the project results in the form of a certification scheme. However, it can be predicted that this strategy will follow the approach proposed by the newly created European Organization of Testing and Certification (EOTC). This organization, by the means of the European Committee of Information Technology Certification (ECITC), promotes the establishment of "recognition arrangements" between test laboratories, certification agency, user and producer organizations at the European level. The purpose of a "recognition arrangement" is to define a certification scheme that allows the mutual recognition of test reports and certificates in order to avoid the cost of multiple evaluations and certifications. In order to be accepted by all parties involved, the scheme must enforce fairness and reproducible results. This implies the constraints of defining understandable public test procedures. In consequence, the scheme must be based on largely accepted standards, at international or European level. Moreover the test procedures must be as objective and deterministic as possible.

When a certain amount of subjectivity cannot be avoided (expert opinion), the test procedure should be strongly constrained by guidelines.

2.1.1.2. Technical Constraints

The constraints listed above exist for any European wide certification scheme. Let us now consider those specific to software. In SCOPE, we aim at assessing the conformity of the software product to specified requirements. Any certification scheme based on this approach would have to describe, in its technical procedure, how these requirements are developed or identified (see figure 1).

It is very likely that the requirements have two origins: the first one is the certification scheme and concerns general requirements applicable to any product, and the second one is the product to be assessed and describes specific requirements. The first step of any assessment case will be to aggregate the requirements coming from these sources.

Once the requirements have been defined, the technical procedure must specify how to conduct testing in order to assess the conformity to these requirements. According to SCOPE philosophy, testing consists, as much as possible, in measuring product components. Here again, the specifics of software industry must be taken into account. To require the use of a specific method to develop software products would indeed be too rigid. It is, therefore, not likely that the constituents of products submitted to assessment will always be comparable. The assessment method must take this constraint into account.

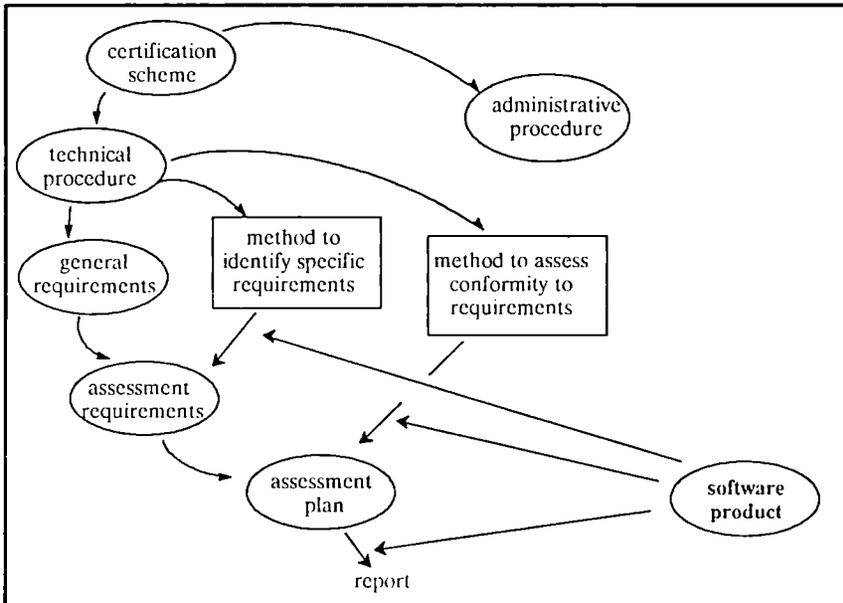


Fig. 1: Technical Aspects of a Certification Scheme

These considerations impose the structure of the assessment framework proposed by SCOPE:

- a strategy to state the requirements to which the product must conform,

- a strategy to identify product components,
- a method to assess product conformity to the stated requirements.

In this context, the technical aspects of a certification scheme are twofold. The general requirements imposed to any submitted product must be identified and the approach used to identify the requirements specific to each product is to be defined. The main effort, so far, has been concentrated on the definition of the assessment framework. Its structure is presented above and its components are described in more details in the following sections

2.1.2. Method/Approach to state requirements

The requirements to which the product should conform must be stated before any assessment can take place. They are usually partly derived from the requirement specification that may have been produced at the beginning of the product development project.

The fact that it is mandatory to enforce the assessment process understandability and clarity imposes to restrict what can be specified as assessment requirement. As opposed to development process requirements specification. Where any type of requirement may be expressed, the assessment requirements must be described in an unambiguous framework.

Another aspect to be considered, is that conformity to the stated requirements must be a piece of information useful to the certification scheme clients; the requirements should therefore be significant of their perspective of quality. The possibility to state non functional requirements must be available.

In order to respond to these needs, we have chosen to limit the assessment requirements to six characteristics. The choice is compatible with the Draft International Standard DIS 9126 developed by the joint technical committee of ISO and IEC (ISO/IEC/JTC1/SC7). These characteristics are: functionality, reliability, efficiency, maintainability, portability and usability (see fig. 2).

Functionality	Maintainability
Reliability	Portability
Efficiency	Usability

Fig. 2: Software Characteristics in DIS 9126

The purpose of this section is not to describe these characteristics in detail but to present their use to state assessment requirements. For each of them, it should be made reference to appropriate requirements document. For example, in the case of "functionalities", a description of the intended functions of the product (a user manual or a more formal functional specification) will be referred to; in the case of "portability", the set of hardware and operating system platforms on which the portability is to be achieved, must be listed.

Of course, depending on the case, some characteristics may be omitted. Moreover, provision has been made to allow to define levels of assurance in the conformity to the required characteristics. It is likely that these levels will be used, in a certification scheme, to state general requirements, while the product specific requirements will consist in descriptive documents referred to.

2.1.3. Method/approach to identify product components & process evidence

Depending on the requirements stated (and on the level required), different components of the software product might be needed, as well as information about the way it has been made. Since we do not want to impose any specific development method, we must be able to use the deliverables specified by those used in industry now and (hopefully) in the future. We have thus developed two description schemes aimed at recognizing the elements of software products and development processes. The aim is not to identify all the elements but only those that are useful in the assessment process.

These description schemes are presented in detail in the SCOPE report R 1.1.1.2 "Certification Model - Definition Report" [6].

2.1.4. Method to assess the conformity to the stated requirements

To be coherent with both the SCOPE philosophy (to be based on "state of the art" techniques) and the constraints imposed by the aim of setting up a certification scheme, we propose the concept of "Assessment Brick" [7].

This concept is based on a modular approach of assessment technology. An "assessment brick" is an object that describes the application of an evaluation technique to identified product parts or development process evidence, in order to measure a specified quality characteristic. The "brick" concept can be considered from two viewpoints.

The first one is formal where the brick is a relation between a product model, a process model, a characteristics model and an evaluation techniques model. The second one is informal and the "brick" is a document that consists of two parts: the interface description and the application guideline. The interface description provides the necessary information to decide the brick application, while the application guidelines contain the expertise needed to apply the assessment techniques (see figure 3).

The bricks are gathered in a library and are used to specify the assessment plan aimed at assessing the conformity of a software product to quality requirements. The elaboration of this plan takes into account the characteristics to be assessed, the required level for each of them, the available product parts and the applicable techniques.

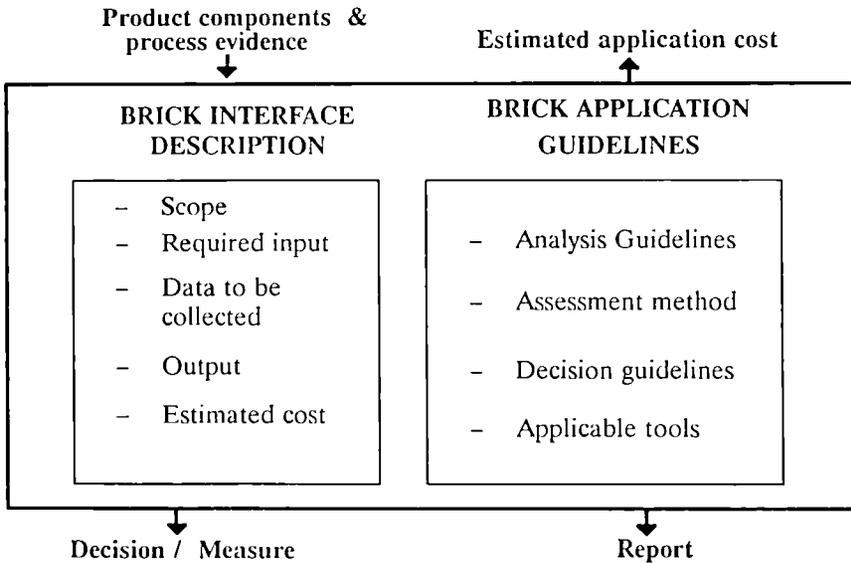


Fig 3: The Brick Document

On completion of the plan execution, a report is produced. It contains elements concerning the plan elaboration and execution and the synthesis of brick reports. This report can be used by a certification agency to decide whether a certificate is to be granted or not.

2.2. Practical Experience

To obtain practically applicable results, the SCOPE project has been organized around two case study campaigns. By case studies we mean software development projects, conducted outside SCOPE, and monitored by SCOPE partners. Monitoring consists in putting in practice all the actions specified by our proposed assessment method. The primary goal of case studies is to assess the practical applicability of the SCOPE approach.

In the current stage of the project, the first case study campaign has been terminated, while the second one is just being started. One aspect of the first set of case studies has been to build, in a bottom-up approach, the elements of our assessment method. However, most important has been the experience that we have acquired on the practical issues of software assessment. There are, indeed, difficult issues such as relationship with the data owners, economic aspects or data management.

The experience gained during the first set of case studies and the planning of the second set is of great value with regard to the above mentioned issues. Most of them concern the technical issues of conducting an assessment.

2.2.1. First Set of Case Studies

A first set of six case studies has been performed during the first 18 months of the project. The goal was to identify, from a practical point of view, the technical issues of software assessment. Their major outcome was the outline of the currently proposed assessment framework, including several assessment bricks. In addition to these proposals, special effort has been dedicated to solve practical issues of data collection

and management. The assessment techniques experimented are indeed producing a lot of data that need to be processed with much care.

2.2.2. Practical Issues of Software Assessment

The need to run certification experiments raises a certain number of issues. The first one consists of economic aspects. It is clear the any assessment activity is very costly and that demonstrating the cost effectiveness of the proposed approach is a key issue. The strategy used in the case studies has been to take advantage of existing tools that could help automate assessment procedures. Furthermore, small prototype transfer tools have been developed to gather and collect data in a central data base for analysis. The problem of data accuracy, when manual data collection is not avoidable, has been decreased by the use of on-site computerized forms or checklist managers. Finally, in order to solve the problem of compatibility of data coming from different sources, a standardized metric format has been developed.

Another issue concerns the relationship that needs to be set up between a case study performer and a provider. A problematic aspect lies with the data confidentiality. To solve these issues, we have limited the connections between the case study provider and the project. One partner acts as a single connection point and, when required, even the identity of the provider is unknown to the rest of the project; this partner and the provider can set up a confidentiality agreement. This experience in setting up relationships between partners and providers is used in the context of the legal study to develop contractual material for certification.

2.3. Legal Study

Certification is an administrative act that has strong legal meaning. On the other hand, the legal status of Software is not clear yet and it is extensively debated at the moment. This explains the importance of legal aspects within SCOPE and their influence on the technical issues of software assessment. The effort involved in the legal study is significant (6% of the total effort).

The aim of this study has been to investigate the legal aspects of software and of certification. This study should cover the legal systems of several EC member states in order to identify the differences that could make it difficult to set up a European wide certification scheme for software products.

The main achievements so far have been to produce an overview of legal issues relating to software liability and certification, a detailed analysis of United Kingdom and French legal provisions with specific reference to:

- legal status and significance of product standards,
- legal status of certification of products,
- liabilities of parties engaged in the certification processes,
- relationship between certification and general provisions relating to software liability.

The French and United Kingdom legal systems were selected for detailed study as representatives of the diverse civil and common law traditions. The work conducted has enabled the core issues relating to the legal aspects of certification to be identified. These indicate that no insuperable legal obstacles exist which might impede the introduction of a community wide certification scheme although national provisions do

contain significant variations, most notably in relation to the acceptance of contractual and non-contractual exclusion clauses.

These issues are being addressed in relation to the other EC Member States. This has been prepared by developing detailed questionnaires and looking for legal case studies.

In addition, the applicability of software certification has been studied by the production of draft contractual documents to assist in determining the relationship between software producers and certification agencies.

3. Current Work and Perspectives

The definition phase of the project is now terminated and we have started the second one dedicated to experimentation and study of implementation issues. The emphasis is put on case studies, on the one hand, and on standardization, on the other.

3.1. Case Studies

As already expressed above, case studies are vital to demonstrate the technical feasibility of software certification. Of course, the case studies performed within SCOPE do not aim at delivering a label. They are rather oriented toward the experimentation of the SCOPE assessment framework in order to get practical results on its applicability.

In order to achieve these aims, we have looked for a wider implication of test laboratories or of consultancy professionals. This was done by enlarging the partnership of the project after the first 18 months. All the new partners and associate partners are performing case studies and contributions are expected from them on the understandability of the framework. Together with the other experts within the SCOPE consortium, they will be able to provide statements on the pertinence of the proposed assessment procedures. Hopefully, improvements will be suggested.

The other expected result from SCOPE case studies is precise data concerning the economic aspects of software assessment. From the first campaign, we already have significant information about the cost of applying some techniques. We are gathering data to complete these results. In addition, the development of small interface tools allows us to use commercial packages to perform various assessment in a very economic way.

3.2. Standardization

The major objective of SCOPE is to demonstrate the applicability of software certification. Since any such certification scheme is likely to be based on a proper standard, we need to investigate the possibilities of transferring our result to standardization.

Since the beginning of the project, liaisons with several standardization bodies have been set up, both at national and international level. This allowed us to understand more clearly the organization of the standardization process. A strategy has, therefore, been designed to influence significantly standardization work to transfer SCOPE's findings.

3.2.1. The Standardization Arena

From global point of view, standardization can be seen at three levels: National, European and International. At the European level, three bodies exist CEN (Comité Européen de Normalisation), Cenelec (Comité Européen de Normalisation Electrotech-

nique) and ETSI (European Telecommunication Standard Institute). They are mapped on the international standardization bodies: ISO (International Standard Organization), IEC (International Electrotechnical Commission) and CCITT (Comité Consultatif International des Télégraphes et Téléphones). In the field concerned by SCOPE, the relevant international committee is the joint technical committee of ISO and IEC (JTC1). At the European level, that field is not covered, although some recent initiative, the CEN Project Team PT 003 [8], indicates that this situation might change.

The national standardization bodies (AFNOR, BSI, DIN, AENOR, for example) are members of CEN (for EC and EFTA), of ISO and of JTC1. The standardization work makes progress by contributions of individual experts, but the decisions are taken by votes of the representatives of the national bodies. This is the reason why the standardization work at international level is often considered as very slow.

3.2.2. SCOPE Approach to Standardization

In the current context of standardization, the only practical way of influencing the development of standards is to participate to committee discussions. To do so, several SCOPE partners have joined their national standardization body and are participating in JTC1 discussions at the international level.

The approach is twofold. It is of great importance to follow on going work in order to feed newly developed standards into the project and therefore avoid incompatibilities between SCOPE and international standards. In addition, we work to promote the adoption of work items to which contributions can be made with SCOPE material.

Another important issue in our standardization strategy is the liaison we have established with other ESPRIT projects dealing with quantitative aspects of software engineering: METKIT, AMI, PYRAMID, COSMOS for example. Regular meetings are organized in order to harmonize the positions of the various projects.

3.3. Expected Results - Conclusions

The SCOPE project started more than two years ago and it is now possible to foresee precisely what kind of results are to be expected. These results, that will be available at the beginning of 1993, can be folded into two categories: those that will directly contribute to the setting-up of a certification scheme and those that may have a broader impact on software industry.

3.3.1. Setting up a Certification Scheme for Software Products

From a technical point of view, the most important result produced by SCOPE will be a concrete, coherent set of evaluation procedures. These procedures, based on direct measurement of the software product components (user documentation, source code or test cases), consist of standardized metrics definitions together with the formalized expertise needed to interpret them. They cover a wide spectrum of software quality including characteristics like functional correctness, maintainability or usability and a framework is provided to choose and integrate them in order to assess the conformity of products to their specific requirements. They are largely based on automatic data collection mechanism using commercially available tools. A preliminary version of this set already exists and is extensively experimented on real life case studies.

The integrating framework, together with the evaluation procedures is a valuable material that will be discussed in the appropriate international standardization committees. As such, SCOPE work influences standardization. Alternatively, on going standardization work is taken into account in SCOPE in order to avoid future incompatibilities.

Finally, the legal aspects of certifying software will be clearly understood, especially with the draft contracts describing the potential relationships between software producers, assessors, certifiers and users. These results will cover the majority of EC member states.

3.3.2. *Impact of SCOPE on Software Industry*

Of course, the setting-up of a scheme for certifying software products will have a definite impact on software industry. However, some more direct results can be expected. The standard measurement procedures proposed by SCOPE are usable outside the context of independent assessment, in the implementation of quality plans for example. The fact that these procedure are standardized will help to their dissemination and to promote the use of quantitative approaches. This influence on software industry is coherent with the aims of other ESPRIT projects such as METKIT, AMI or PYRAMID.

Acknowledgements

The author would like to thank Alain Roan for his useful comments in writing this paper, as well as all the participants in SCOPE for their contributions to the results presented here.

References

- [1] P. Robert and A. Roan, The SCOPE Project: an Overview, in W. Ehrenberger (editor), Approving Software Products, Elsevier Science Publisher, September 1990.
- [2] SCOPE Consortium, Software Certification: State of the Art, SC.89/066/SCOPE/R.1.1.1.1/RL/01, September 1989.
- [3] SCOPE Consortium, Software Certification: The European Context, SC.91/VLG.phr/R2.6.3.1.1/01, February 1991.
- [4] F. Rienstra, Software Quality Assurance, In the CTS Technical Days, CEC, Oct. 1987.
- [5] Dr. G. Knorr, The Gütegemeinschaft Software - A major Concept in the Certification of Software Quality, in W. Ehrenberger (editor), Approving Software Products, Elsevier Science Publisher, September 1990.
- [6] SCOPE Consortium, Certification Model, SC.90/031/ECT.jb/R.1.1.1.2/RL/05, August 1990.
- [7] P. Robert and F. Seigneur, A Modular Approach for Software Product Assessment: the BRICK Concept, In proc. of Eurometrics'91, EC2, March 1991.
- [8] Comité Européen de Normalisation, Study and Investigation related to Information Systems Engineering, PT003/BC-IT-014 SI, May 1991.



THE SPIRIT HIGH PERFORMANCE TECHNICAL WORKSTATION

R. WIECZOREK* (D), R. GRIMSDALE (GB), P. LECLERC (F),
M. DE LANGE (NL), W. STRÄBER (D), D. BEAL (GB)

*The SPIRIT Consortium
KONTRON Elektronik GmbH
Oskar-von-Miller-Str. 1
8057 Eching bei München
tel: + +49 89 319 01 344
fax: + +49 89 319 01 575

SUMMARY

This paper presents the SPIRIT High Performance Workstation. The SPIRIT Hardware Architecture, High performance Graphics Subsystem and Operating System are discussed, concluding with a discussion on some examples for applications using the SPIRIT Workstation.

1. INTRODUCTION

SPIRIT is a high performance technical workstation designed specifically for advanced applications in engineering design, medical image processing and scientific visualisation. The SPIRIT Workstation combines exceptional computing power using a multiprocessor architecture with high performance 3D real time, true colour graphics.

SPIRIT is a major project funded by the European Strategic Programme for Research in Information Technology (ESPRIT) which will put Europe in a strong position to compete in a market currently dominated by the USA and Japan.

The SPIRIT Workstation is a tightly coupled multiprocessor design, communicating over a bus with a sustained bandwidth of over 100 MBytes per second. It provides 80-600 MIPS and 20-140 MFLOPS (double precision LINPACK), with data rates capable of transmitting realtime video and providing real-time 3D graphical interaction. The software base is an industry standard UNIX system, augmented with object-oriented programming environments and AI languages, all fully integrated. The system software is implemented with advanced software technology which includes a multi-threaded, tightly coupled, heterogeneous operating system kernel. This support for heterogeneity is the key prerequisite to incorporate new processors as they appear. The SPIRIT Workstation is equipped with standard networking facilities.

The industrial partners will market commercial products to establish a dominant position in the high-performance end of the European workstation market by the mid 1990s.

2. SPIRIT HARDWARE ARCHITECTURE

The SPIRIT Workstation is a single user superworkstation with a very high speed graphics accelerator. It brings the power and performance of a mini-supercomputer but with lower price and excellent interactivity.

The SPIRIT Workstation achieves much higher performance than single processor workstations. Its Multiprocessing technology based on advanced microprocessors sustains a performance of 80-600 MIPS and 20-140 MFLOPS (double precision LINPACK) under a UNIX operating system.

No special code optimisation for vector processing is needed since the floating point units have a scalar architecture. The application programs make use of multiprocessing by means of parallel multitasking, i.e. the application is split into several threads which are automatically distributed amongst the processors by the UNIX operating system.

The SPIRIT hardware consists of several boards connected by high speed buses. Figure 1 gives an overview:

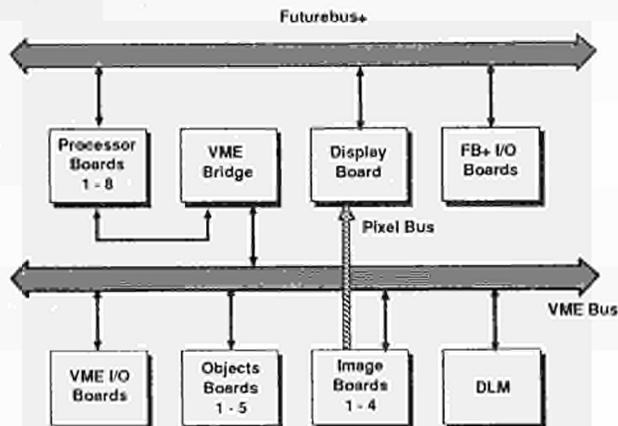


Fig. 1: Architecture of the SPIRIT workstation

The basic hardware components developed at KONTRON Elektronik GmbH are the Processor Boards, the VMEbus Bridge, the Display Board, the I/O boards and the bus system. The Declarative Language Machine (DLM) was developed at British Aerospace. It is an Artificial Intelligence computer which accelerates PROLOG. The High Performance Graphics Subsystem is being built by CAPTION in collaboration with QUEEN MARY AND WESTFIELD COLLEGE, the UNIVERSITY of SUSSEX at BRIGHTON and the UNIVERSITÄT TÜBINGEN (section 3).

Bus System

To decouple data traffic of memory accesses, I/O, and pixel data, several standardised buses are used in the SPIRIT Workstation. In addition to the gain in speed of this concept, these buses introduce compatibility to many hundreds of off-the-shelf boards.

The Futurebus+ is used by the processors to access the shared main memory. It allows for cache coherence between the copyback caches of the processors. Bandwidth of main memory and the Futurebus+ (more than 100 MByte/s sustained) is sufficient to connect 4 to 32 processors without saturation. The Futurebus+ interfaces of the boards are designed to interoperate with the profile F+ (Profiles are defined by the Futurebus+ committee to describe a set of options that ensure compatibility).

The VMEbus is used as a local bus for the High Performance Graphics Subsystem, (section 3). The VMEbus has been on the market for years and is very well accepted by the industry.

It permits high speed I/O boards from other vendors to be incorporated into the system. It is also used to connect the standard I/O subsystem of the SPIRIT Workstation.

The Futurebus+ I/O boards contain S-Bus interfaces. They are used to connect I/O modules compatible with the I/O modules of the SPARCstation (Sun Microsystems).

A separate pixel bus is used as a local connection between the Image Boards and the Display Board. Its high data rate (200 MByte/s) is needed to output screen images at up to 25 frames/s. This is sufficient for true colour animated images.

Processor System

The Processor Boards of the SPIRIT Workstation are based on the MC68040, CISC processors from Motorola. Four processors will be located on each of the 1-8 Futurebus+ based Processor Boards. The first versions of Motorola's MC68040 processors run at 25 MHz clock speed. The Processor Board is designed to work at clock rates up to 40 MHz without any changes. This will allow the performance to be scaled up when Motorola delivers processor chips at higher clock rates.

The Processor Board (figure 2) contains a Futurebus+ interface, a DRAM controller with a memory array, four MC68040 processors with second level caches and a connection to the VMEbus bridge.

Main memory can be 16-256 MByte per board, i.e. up to 2 Gbyte per system. Sustained memory bandwidth is 100 MByte/s at 25 MHz clock speed, scaling up to 160 MByte/s at 40 MHz clock speed. The memory can also be accessed from the Futurebus+ using the same physical addresses. Thus, the memory of all processor boards is shared (global) amongst the processors.

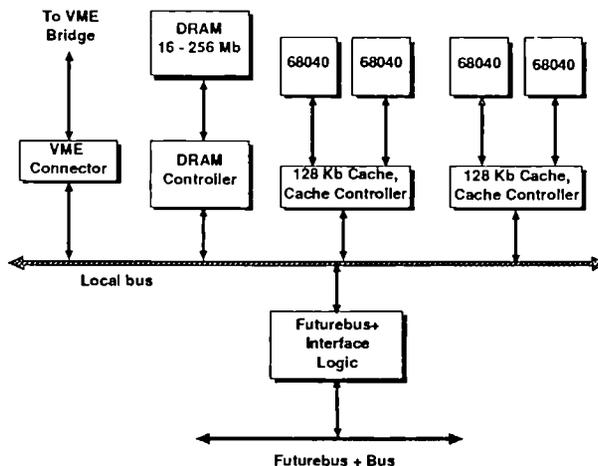


Fig. 2: Futurebus+ Processor Board

VMEbus Bridge and VME I/O Boards

The VMEbus Bridge connects the internal bus of one processor board to the VMEbus. This bridge can also be accessed from the other processor boards by means of the Futurebus+. Accesses from VMEbus masters to the main memory maintain cache coherence using the cache incorporated in the VMEbus Bridge. This cache also significantly reduces bus bandwidth consumption of DMA accesses. Accesses from the processor boards to the VMEbus are performed without cacheing.

The VMEbus bridge is also used to connect the VME I/O boards as well as the Object and the Image boards. Peripheral interface modules are available for SCSI devices (disk, tape, etc), floppy disks, Ethernet, multiple serial lines and for a disk array.

Futurebus+ I/O Board

Compared to the VMEbus I/O boards the Futurebus+ I/O board (figure 3) will offer higher performance and will use a standardised mezzanine bus (s-Bus) for the peripheral interface modules.

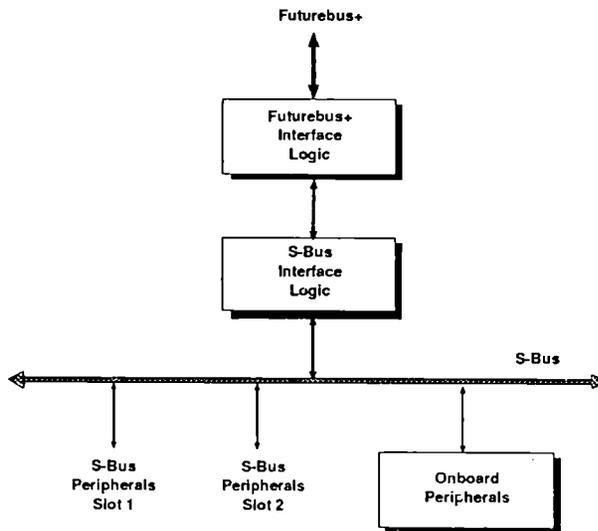


Fig. 3: Futurebus+ I/O Board

The Futurebus+ I/O Board will contain several onboard peripheral interfaces, Floppy disk controller, EPROM, audio controller (ISDN), Real Time Clock, battery backed RAM and SCC (serial interfaces). It is planned to develop S-Bus interface modules for FDDI, SCSI disk array, Ethernet, X.25/ISDN and 16 channel RS232. These will plug into the two S-Bus connectors residing on the main I/O board. For FDDI, Ethernet and SCSI disk array, fast 32 bit wide I/O controller chips from National Semiconductor and NCR will be used, which offer the best performance without sacrificing much bus bandwidth for I/O. A cache is used to maintain cache coherence for I/O while bus bandwidth significantly.

3. HIGH PERFORMANCE GRAPHICS SUBSYSTEM (HPGS)

The HPGS (figure 4) provides the ultimate in high quality 3D interactive graphics. For users only requiring 2D graphics, the SPIRIT Workstation can be provided with just the Display Board. The HPGS architecture comprises three board types over which the conceptual graphics pipeline is mapped:

- The Object Boards provide high computational power for geometric processing and management of the graphics database.
- The Image Boards provide high computational power for very fast triangle and vector shading with depth buffer and anti-aliasing.
- The Display Board provides a double buffer frame store and display technology.

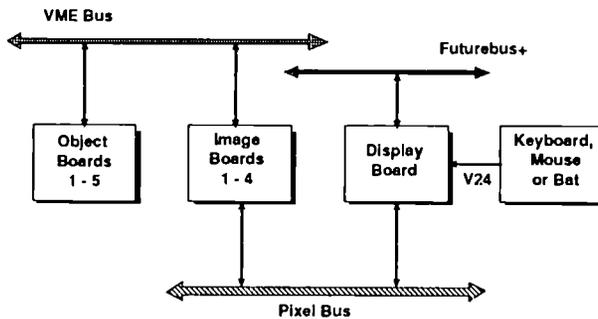


Fig. 4: Architecture of the SPIRIT workstation HPGS

Because of the scalability afforded by the bus system (Futurebus+, VMEbus, Pixel bus) and the flexibility of the Graphical Interface Layer (GIL), (section 3), it is possible to configure the HPGS in a number of options ranging from a low cost graphics subsystem consisting of just the display board to the full blown graphics subsystem consisting of five object boards, four image boards and a display board providing 2D and 3D interactive graphics in real time. Performance of the HPGS is:

- 300,000 Gouraud shaded triangles per second.
- One million vectors per second.
- Phong shading at 100,000 pixels per second.

The Intel 860 Processor Module

The i860 processor forms the processing elements common to both object and image boards. Figure 5 illustrates the architecture of the i860 module.

The i860 module is composed of an i860 processor, up to 8 MBytes of local memory, an interrupt and timing controller.

The Object Board Architecture

The object board architecture is composed of five i860 processors, shared memory (called local memory in figure 6), communication memory, local and communication buses. The i860 processors are configured as one master with four slave processors using standard master/slave bus arbitration protocols.

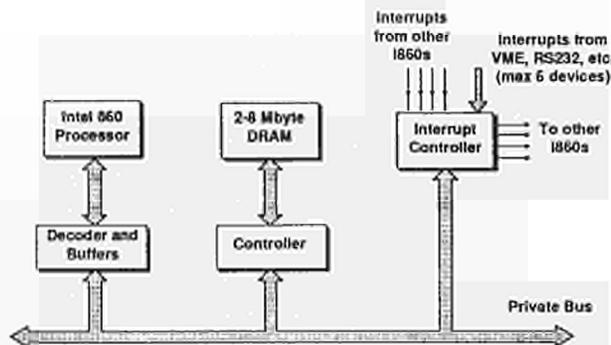


Fig. 5: Architecture of the SPIRIT workstation i860 module

The master processor controls the communication between the object board and all other VMEbus subsystems. It manages the traversal and flattening of the hierarchical database and farms out work to the slave processors. The slave processors provide the computational power for geometric and lighting computations, processing of NURBS, etc. The Object board also performs high quality rendering such as ray and radiosity. The output from the object board(s) is sent to the image board for scan line rendering.

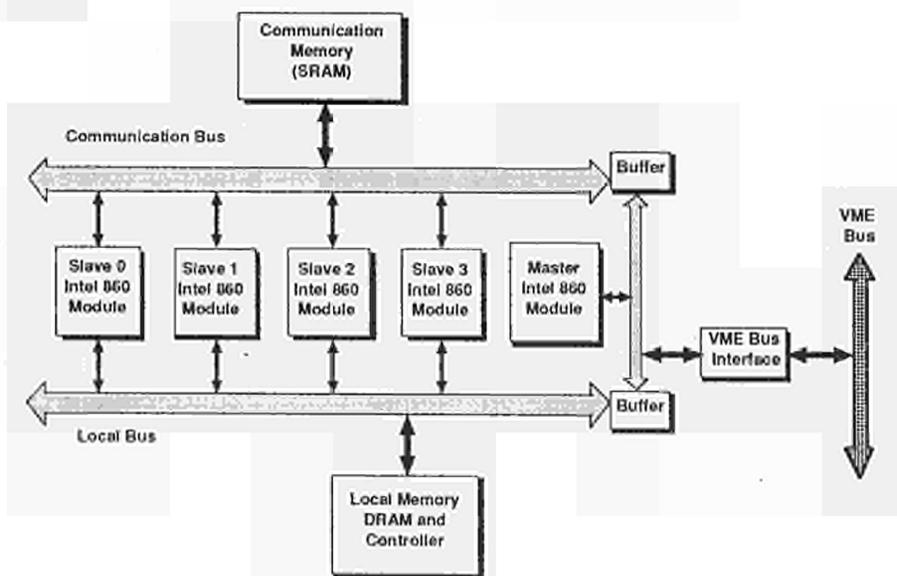


Fig. 6: Architecture of the SPIRIT workstation object board

The Image Board Architecture

The image board architecture (figure 7) features two i860 processors, DMA and I/O logic, image and depth buffer and VME interface. The i860 processors are used to compute and set up parameters such as depth and colour gradients for Gouraud shading with the ASIC Drawing Engine (ADE).

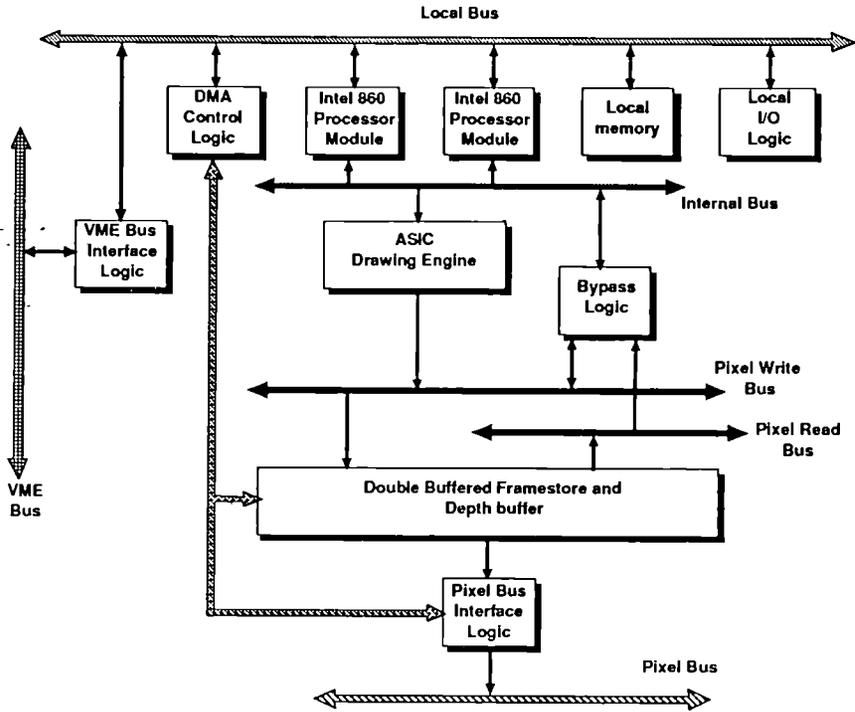


Fig. 7: Architecture of the SPIRIT workstation image board

ASIC Drawing Engine (ADE)

The ADE is a custom VLSI device used to accelerate Gouraud Shading with anti-aliasing which can also act as a pixel generator for Phong shading algorithms. The ADE scan converts triangles and vector primitives using the Pineda and Bresenham algorithms, respectively. The ADE outputs Gouraud shaded pixels with anti-aliasing information once every clock style. In particular the ADE scan converts the following geometric primitives:

- Wireframe triangles
- Hidden-line removal triangles
- Gouraud shaded triangles with or without anti-aliasing
- Anti-aliasing vectors with colour and depth interpolation.

HARDWARE ARCHITECTURE

The ADE (figure 8) is split functionally into eight units: state machine and control/timing logic, loading logic, geometry computation, windowing computation, depth computation, colour computation, anti-aliasing computation and output formatting. Besides its unprecedented speed, the most remarkable feature of the ADE is its unique highly efficient anti-aliasing scheme.

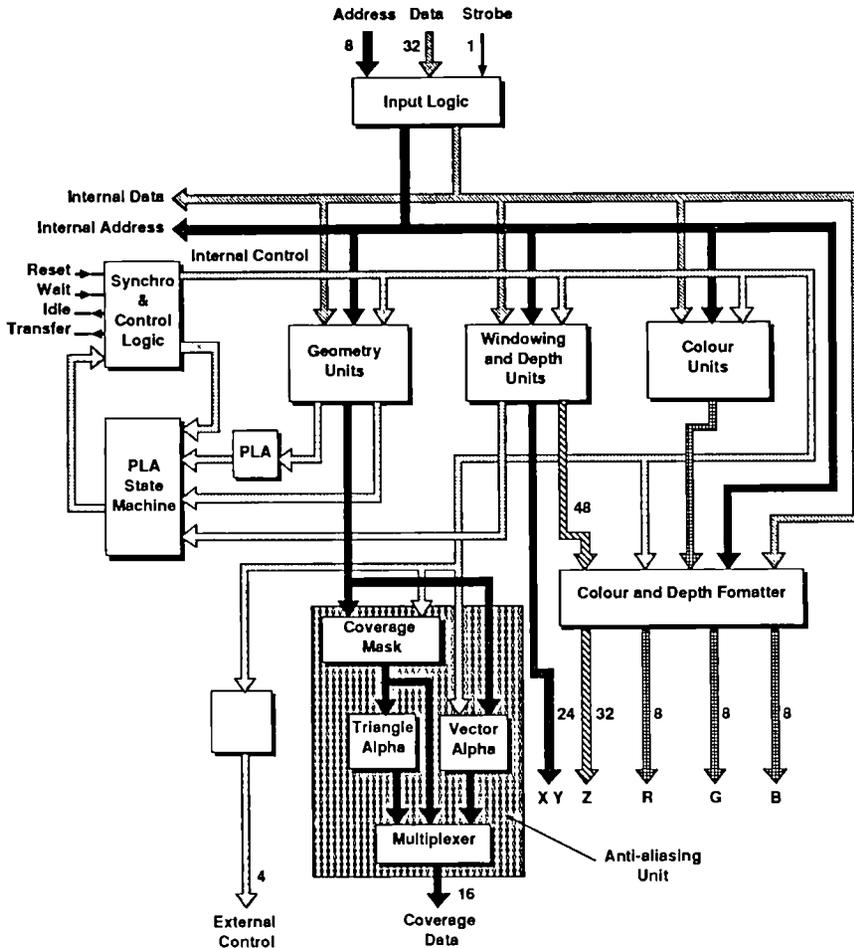


Fig. 8: Architecture of the SPIRIT workstation ADE

Display Board

The Display Board (figure 9) is a high performance video output device with a fast 2D graphics processor. It provides a Futurebus+ interface, true colour RAMDAC and video memory, a graphics processor with shared DRAM and interfaces for mouse and keyboards. A high performance pixel bus (200 MByte/s) connects it to the Image Board(s).

The TMS34020 graphics processor is used for 2D graphics computation, e.g. windowing. It supports the low level features of the 2D GIL (Graphics Interface Layer) including colour palette handling, cursor movement, mouse and keyboard control.

An 8 MByte frame buffer is used to store the image. This frame buffer is organised as 24 bits of image true colour information, 4 bits of overlay, and 4 bits to select one out of 16 colour palettes; 8 MBytes of additional memory may be used for double buffering. Screen resolution up to 1600 x 1200 pixels and refresh rates up to 75 frames/second (non-interlaces) are supported. An 8 MByte internal DRAM is used for

display list memory and as the TMS34020 workspace. The RGB signals are produced by a RAMDAC true colour lookup table chip (Brooktree 463).

It supports true colour as well as pseudo colour and the X Window system.

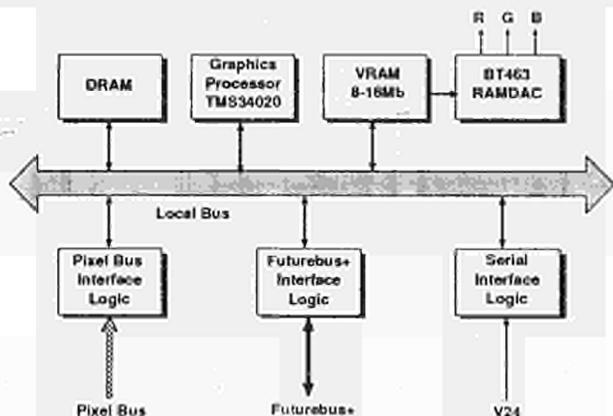


Fig. 9: Architecture of the SPIRIT workstation display board

The Graphical Interface Layer (GIL)

The GIL was conceived as a device independent interface to the HPGS, supporting both the HPGS and the option in which only the display board is installed. This GIL is basically composed of three layers:

1. The Structure Interface Layer (STIL) provides the interface between clients and the rendering layers for the creation of hierarchical scene descriptions with transformation matrices, traversal mechanism, etc. PHIGS, PHIGS PLUS, RenderMan, etc. would be ported to the SPIRIT Workstation at this layer.
2. The 3D Layer provides the interface between the object and image board hardware and software clients such as PHIGS PLUS, RenderMan, 3D applications, etc. The 3D layer uses some of the concepts of the 2D layer.
3. The 2D Layer provides the interface between the display board hardware and software clients such as PHIGS PLUS, etc. which require support for 2D primitives.

4. SPIRIT Operating System (OS)

The kernel of the OS for the SPIRIT Workstation has been designed and implemented to meet the primary objective of the SPIRIT system, the highest performance possible for a single Workstation. This is why both the hardware architecture and the OS kernel software of the SPIRIT Workstation are developed around a tightly couple shared memory multiprocessor concept, where shared means that the memory is directly accessible to all processors, and not emulated at the expense of significant speed reduction as is often found in network oriented systems. This approach allows all processors in the system to access shared data resources at the maximum possible speed.

The second key requirement for the SPIRIT Operating System is the strict adherence to standards in order to allow quick and easy porting of applications. To this end the SPIRIT kernel has been designed and developed as a symmetrical multiprocessor

version of the AT&T System V UNIX implementation. In this way, both the top-performance and standards requirement are met, and SPIRIT Workstation users are provided with a high speed system V platform that additionally offers X/Open and POSIX compatibility, a BSD environment and the appropriate BCS's for the processor types supported in the heterogeneous operating systems.

The heterogeneous nature of the SPIRIT Workstation allows processors of different types to operate on this single, fast access shared memory, in a way that is fully transparent to the SPIRIT user.

The portability and heterogeneity of the SPIRIT kernel make it unique and represent an advance over symmetrical kernels developed by others such as Sequent. Moreover, the SPIRIT Workstation OS is a forerunner of such current developments undertaken by the Intel Multiprocessor Consortium which in any case do not address the heterogeneity aspect at all.

It has a continuing consideration in the design and development of the SPIRIT kernel that to provide speed, heterogeneity and optimal usage as a technical workstation are the prime criteria for the markets at which the SPIRIT Workstation is targeted.

Shared Memory

The multiprocessor kernel is designed for memory sharing and for performance reasons avoids internal message passing. It is structured using a microkernel concept, and critical data are protected and synchronised using monitor and region primitives. The SPIRIT kernel is engineered to allow optimal parallelism which comes cheap because access to resources in SPIRIT is guarded through low cost synchronisation primitives.

Even though some promising research has been done by CMU and others on distributed shared memory, the distributed systems used today still suffer significantly from lower performance when compared to shared memory systems such as the SPIRIT kernel as a result of message passing overheads, while on some of such systems the distributed UNIX semantics are offered based on a single threaded server lacking any parallelism.

The hardware architecture of the SPIRIT Workstation processor boards are based on the Futurebus+ which supports the shared memory model through coherent caches. As a consequence contention is reduced to a minimum.

Each processor runs one instantiation of the microkernel that maintains a very small amount of private memory for local administration purposes. All other memory appears in one physical address space and is shared by all processors, while access is classified according to its usage and its type of synchronisation through a hierarchy of monitors and synchronisation primitives. The data resources of the kernel are common to all processors and are shared by all kernel processes, and by user processors when these latter perform system-call operations in the kernel.

User processes, when executing user level code, share any of their segments with other processes. This allows parallelised applications such as a database application or the SPIRIT Multiprocessor SmallTalk implementation, to efficiently share common data or code.

The versatile segment sharing facilities extend the standard UNIX segmentation by provisions for arbitrary numbers of segments, programmable attributes such as access types, size, inheritance and sharing. These facilities are basis of the SPIRIT implementation of threads. Threads, or light-weight processes, in the SPIRIT OS are user processes that share most or all of their segments. However, they are full UNIX

processes implying that unlike threads in many other implementations, a thread in SPIRIT can perform real asynchronous, parallel and autonomous operations like blocking on input or output while leaving its fellow-threads undisturbed. In the unlikely case that two threads of the same process run on the same processor, context switching overhead between them is minimal because of the sharing of their address space.

The basis of the UNIX Operating System for the SPIRIT Workstation is a multiprocessor implementation of UNIX. One of the requirements for the multiprocessor OS is that it can be ported to various H/W platforms relatively easily. To this end a Hardware Software Interface (HSI) has been specified. It is being implemented for the MC68030 and MC68040 processors and for the VME030, MVME147, and CCCI-processor boards.

Synchronisation and Protecting Critical Data

In order to enhance the performance of the OS by introducing more parallelism in the OS, the microkernel offers better (higher level) mutual exclusion and synchronisation primitives when compared to the monolithic UNIX kernel.

In figures 10 and 11 below circles indicate processes, while rectangles indicate resources. Processes can be anything from interrupt driven small tasks that only perform hardware oriented activities, up to large user applications such as database applications or number crunching programs. On a SPIRIT Workstation several hundred of processes run in parallel and are bound to the group of identical processor types during their life time.

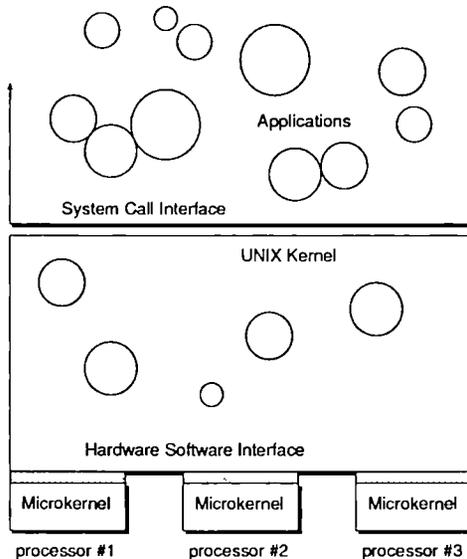


Fig. 10: Layering of the Operating System with Microkernels

The guards for critical data regions and the constructs for synchronisation of processes accessing these are of three different types.

Locked Monitors

Data resources on which the operations take little time when compared to process deactivation time or that should allow direct access from interrupt driven, run-to-completion code, are implemented through locked monitor. These monitors have full 'monitor' semantics and waiting at entering the monitor is performed by spinning for a lock. an event mechanism allows processes to wait in the monitor for a certain condition; such a process then atomically leaves the monitor and is deactivated.

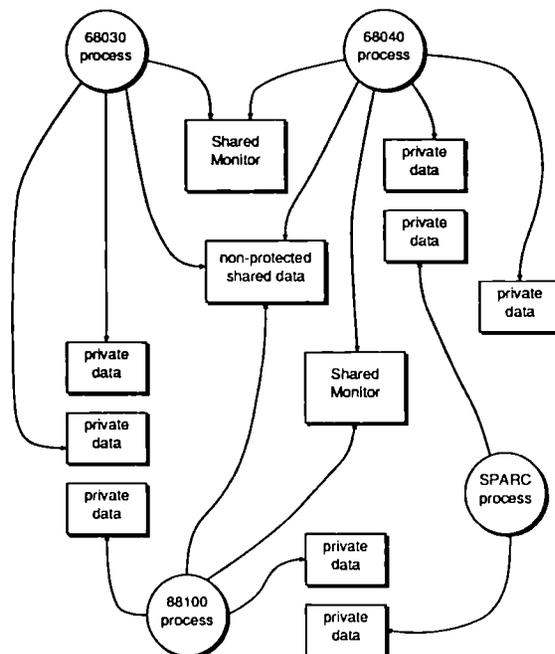


Fig. 11: Processes on Different Architectures sharing Data

Monitors

As most of the concurrency and data resource handling takes place at the process level, 'normal' monitors are most commonly used for operations on resources.

Monitors guard critical data and are implemented using P/V like semaphore operations allowing deactivation and activation of processes concurrently trying to enter the monitor, and allowing processes to wait for a certain condition once inside the monitor. Process activation and deactivation and interaction with the dispatching of the microkernel are again implemented using locked monitors.

Regions

The semantics of the UNIX functionality are such that processes hold resources of the kernel for longer periods, and sometimes resources of the kernel for longer periods, and sometimes resources are even held claimed while the process is deactivated. Some resources claimed when it deactivates, assuming that these resources will be claimed again after continuation. In order to cope with these semantics, and in order to have a generic means to achieve in-kernel parallelism, the region concept was introduced.

Regions are registered with the entering process and offer checkpoint facilities and reclaim-order semantics. A process that deactivates (sleep) will release the regions it possesses and will reclaim these upon activation (wakeup).

The kernel code is carefully divided into independent functional portions that are represented as regions. Since these portions are mostly parts of the UNIX kernel semantics, processes in the kernel run in parallel to a large degree and they are synchronised when they enter the same region and attempt to access the same resource.

Heterogeneity

An innovative aspect of the OS is the concept of heterogeneity in a closely couple shared memory system. This feature has been specified to provide maximum transparency to the user. Any mixture of the above mentioned boards and processors can be present in the final system and cooperate with each other in shared memory.

The heterogeneity and the functional layers that are being shared among processors of different types are visualised in figure 12, which also illustrates how SuperCOFF linkage editor facilities hide the heterogeneity aspects from the programmer/user by putting the various processor dependent instantiations of the same application in one single envelope.

The thin double line indicates the bus of the system. All components *outside* the bus are hardware components while everything *inside* represents software functionality. The exchange-ability of software and hardware for the graphics layers is clearly depicted. Note that figure 12 shows one single system both physically in hardware and logically to the user.

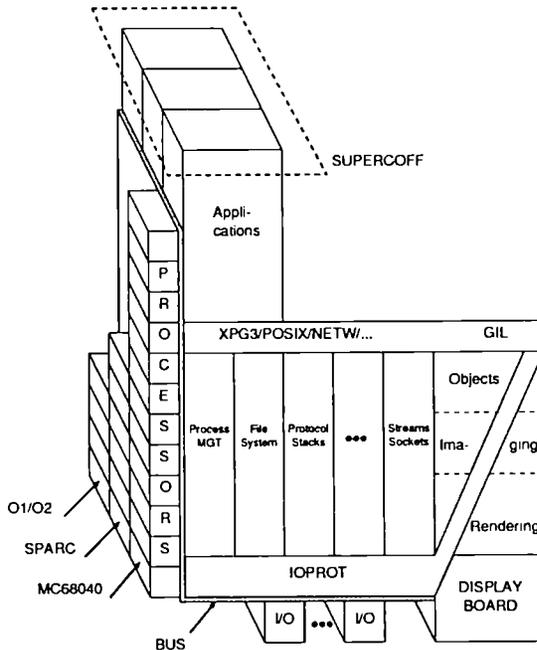


Fig. 12: Everything fitting together in one system

Summary of the SPIRIT Operating System

The Operating System of the SPIRIT Workstation from a user's perspective is UNIX System V. The standard interfaces that the programmers experience are described in the various specifications and comprise XPG/POSIX, BSD4.3, the appropriate OCS/BCS implementations, and the common network and windowing facilities found in the technical workstation area.

The application's interface and the commands and utilities are strictly compliant to the XPG definitions which are in-line with POSIX 1003.2 and are derived from AT&T System V. Since there does not exist a standard programmer's environment for Multiprocessor UNIX various extensions are provided to enable exploration of the Multiprocessor capabilities, e.g. threads and elaborate segment sharing. Other extensions will be real-time capabilities and heterogeneity aspects. The end result is a full-blown standard UNIX with outstanding performance and with a UNIX user environment at least as good and complete as offered by any UNIX system on the market.

For a detailed specification of the OS and the work involved, one is referred to the various specification documents of ACE, in particular the programmer's documentation and notable *Specification Portable Multiprocessor UNIX* which gives a full description of the functionality and implementation choices, and the *System Architecture Ground Rules* in which a pseudo formal definition is given of the restrictions and semantics of the kernel, its processes and operations with respect to access to the shared memory, in particular for heterogeneous operation.

The article *Parallelism and Heterogeneity in the SPIRIT Workstation* that was presented at the May 17, 1990 NLUUG Conference reviews the SPIRIT approach to threads and heterogeneity.

5. APPLICATIONS USING THE SPIRIT WORKSTATION

Automobile Computer Aided Design

The SPIRIT Workstation is ideally suited for the creative activity of car body design. Its high performance allows the designer to develop aesthetic ideas without the impediment of the slow operation of conventional workstations.

The designer first creates a rough impression of the outline shape of a vehicle. He then uses parametric surfaces to get better control of the smoothness of the shape, by picking and modifying some of the control points. Much time is saved since the system automatically recomputes and reconnects the patch boundaries.

3D Medical workstation

One of the most intriguing objectives is the investigation of the shape and form changes of biomedical objects in relation to pathophysiological processes in time. The results combined with knowledge based information from a basis for an expert oriented computer aided decision support system.

In particular, for biomedical computer aided decision support in clinical practice, a geometrically accurate 3D reconstructed model for a realistic and clear-viewed graphical presentation of the surface, and the internal spatial architecture of the composite entities of the organs aided with knowledge based information is needed. Moreover, analysis of the spatial architecture of the various elements and their relation is needed to enable the scientist to identify characteristic features and to describe these features

in quantitative terms in order to understand structure and function. The SPIRIT Workstation is well suited for this 3D medical application.

3D Geometrical Modelling, Visualisation and Analysis

Three dimensional geometric modelling and visualisation for 3D image data analysis as well as decision support algorithms for an objective recognition are needed for the use of medical workstations. For a realistic and accurate geometrical visualisation of the surface of biomedical objects C^1 continuous surfaces using piecewise polynomial interpolation will be important, for which the high speed of the SPIRIT Workstation is essential.

3D image data analysis includes 3D feature identification and extraction. For this purpose object orientation and position computation according to anatomical rules, and computation of three dimensional moment invariants and other geometric measures of the 3D data set are of importance. For a computer aided decision support, expert systems in combination with automated three dimensional pattern (object) recognition techniques are needed to recognise 3D shape form changes of precisely 3D reconstructed biomedical objects.

4D Simulation

Computer simulation of dynamic processes varying continuously in time is one of the great challenges in computer science, e.g. to do growth studies in biology. Characteristic to this method is that it provides morphometric analysis by which shape, form differences and form changes can be determined. It also introduces new computer aided tools to quantify inter- and intra- individual changes in shape and form measures of biomedical elements in relation to their surroundings. The parametric description of 3D shape and morphometric measures of identified object features and their evaluation in time are the basis for computer simulation studies, e.g. morphometric development of organs. Visualisation of this enlarges the understanding of structure and function relationship.

The Expert Architect

The *expert architect* is an interactive system for the production of architectural designs. It makes use of expert knowledge encapsulated in a database. The architect can indicate general requirements for a design and call upon the *expert architect* to add details. The external appearance can be developed whilst the system ensures the consistency of the internal structure. Thus, constraints concerning the size and position of rooms, rules concerning access corridors for rooms and the related positions of windows can be maintained.

The architect can work with several simultaneous displays: a textual description of rules currently being used, a plan of the building and a high quality image of the final structure. The architect can take his client on a *walk through* of the building and changes can be made interactively.

Computer Vision

Computer vision with the aim of 3D reconstruction from 2D views is most promising when following the model driven approach. It assumes enough a priori knowledge is available about all objects of the scene. The reconstruction and recognition process

relies on the capability to match object features on different levels of abstraction between the scene to be identified and the objects in the database. The operations needed are geometric modelling to create instances of generic objects, high performance graphics to display a realistic reconstruction, fast image processing to derive matchable features and feature matching itself. The SPIRIT Workstation incorporates all aspects in a single machine with a unique software environment. Besides the obvious effect of dramatic speedup for computer vision, the more important aspect is the new quality of working environment for the engineer which will encourage new activities.

THE ESPRIT EFFECT ON CIM DEVELOPMENT FROM STRATEGIC TO OPERATIVE LEVEL: THE PIRELLI TYRE HOLDING EXPERIENCE

Giorgio Basaglia
Pirelli Coordinamento Pneumatici S.p.A.
Operations Production
Viale Sarca 222
I-20126 Milano
Tel. XX39.2.64422925
Fax. XX39.2.64424528

Marco Guida
Pirelli Informatica S.p.A.
Expert Systems Group
Via dei Valtorta 48
I-20127 Milano
Tel. XX39.2.28325661
Fax. XX39.2.28324999

1. Introduction

This paper describes the involvement of Pirelli Tyre Holding (PTH) within ESPRIT projects, in which both direct and cooperative participations with other consortium partners have been active since the launch of the Programme.

Particular attention has been given to the EP 2434, currently in its conclusive phase, which plays a fundamental role in a theme initiated over six years ago. Today one can begin to see the results of these efforts whose effects spread throughout all levels of the Pirelli Group, from corporate to operational level at the factory.

The formulation of a comprehensive strategy, with regard to developing advanced IT systems for manufacturing, has been reached via a detailed examination of the organisational and functional context. Indeed, this analysis also provided the basic guidelines for the participation of PTH in European projects and also its subsequent role within ESPRIT, which in so doing permits a better understanding of the relative strong points and possible areas of improvements.

The analysis of software products developed under EP 2434 within the Pirelli Tyre Holding application site, their functions and location within the common reference grid, the industrial benefits both already obtained and awaited, all open the door to new activities and future developments based upon the foundations laid down by these many years of work.

The experience gained over this time allows more general conclusions to be made about the approach and the role which could be taken by an application site in European projects. Thus maximising not only the possible opportunities but, above all, the actual results that could be achieved by all partners involved.

2. Pirelli Group description

The main activities of the Pirelli Group are covered by three Sectors: - Tyres (Pirelli Tyre Holding), Cables and Diversified Products - all of whom report to Pirelli S.p.A.. Pirelli Informatica also reports to a function within this entity.

Pirelli Coordinamento Pneumatici (PCP) is the company within the PTH organisation which controls at corporate level all the activities of the Tyre Sector using the same organisational and functional structure found in the Operating Units (all of whom report directly to PTH).

Within this company, two structures exist in relation to Information Systems; the more traditional IT one covers all the areas of the business, whereas the second one is

dedicated to the specific area of manufacturing systems (under the function of Operations - Production).

These two structures, neither of which develop software, have their counterparts in the Operating Units: the first one is in the IT area and the second is within the Manufacturing Services.

With regard to the management of projects such as ESPRIT, in which Pirelli usually takes the role of application site, the job is naturally found within the coordination center of the Tyre Sector under the function of Operations -- Production - Manufacturing Systems.

In addition, this function must also carry out the integration between the Tyre Sector and Pirelli Informatica (IT company of Pirelli Group); hence the reason why the latter often participates in some projects (like EP 2434) under the common name of Pirelli S.p.A..

Pirelli Informatica develops software products for both the internal and external markets, which entails the commercialisation of resultant packages.

From a completely production oriented point of view, Pirelli Tyre Holding comprises of 22 tyre factories in 10 countries worldwide, eleven of which operate in five countries belonging to the European Community.

3. The involvement of Pirelli Tyre Holding in ESPRIT

The history of Pirelli Tyre's involvement in the ESPRIT projects goes back to 1985, which marks its first participation along with Pirelli Informatica in the EP 932 (Knowledge-Based Real-Time Supervision in CIM).

This large industrial group has always been principally oriented towards production problems or research & development concerning solely product and process. Thus, it was reluctant to become involved in new Information Technology ventures and still further enter into a European Research Project (in its initial stages) with all of the associated risks and without even the positive support of previous results to sustain it. The person responsible for this coordination in Pirelli Coordinamento Pneumatici had to first convince the top management of Pirelli Tyre and then procure the initial commitment of the factories to the programme.

Work started with project partners, following a course of action which now gives results of major consequence, and also provides a base through which the complete success of the ESPRIT endeavours can be confirmed.

Meanwhile Pirelli Informatica started to prepare other ESPRIT projects in addition to the collaboration with Pirelli Tyre. At the same time, other functions of PCP also considered their possible participation in new European programmes and hence the entrance of the Pirelli Group into this new field was marked.

The logical extension of this first step was the subsequent participation in the EP 2434 (Knowledge-Based Real-Time CIM Controllers for Distributed Factory Supervision), the results of which shall be examined in detail in the following sections.

4. ESPRIT Project 2434 and conceptual background

The project aims at using knowledge-based software techniques in order to turn modern production strategies like JIT, OPT, LOP into an operational reality on factory floor. The main goals are:

- Adapting architectures and AIP methods to be suitable for CIM.

- Building expert systems and design tools for workcell and shop controller realisation.
- Testing CIM Controllers for small, medium and large batch manufacturing in running plants of electronic, tyre, cable and IC industries.
- Developing controllers and tools towards marketable products, including dedicated controller chips.

The project is a continuation and logical extension of ESPRIT 932 (Knowledge-Based Real-Time Supervision in CIM) (Meyer and Walters, 1990), and aims to:

- strengthen **vertical integration**: involve more hierarchical levels;
- strengthen **horizontal integration**: more functions involved, including personnel management and procurements in addition to production planning, quality assurance and maintenance;
- **distribute** knowledge-based techniques and shorten reaction times
- extend the consortium and **widen the application sites**.

The main goal of the project is to adopt knowledge-based techniques to support decision makers at the planning and operational levels of the business using software aids like expert systems for interpretation, diagnosis and action planning. In detail, the project aims at bridging the time gap between high level planning systems and the real-time conditions at factory floor, through the development of dynamic scheduling and planning as well as quality and maintenance systems.

The modular system architecture of Workcell and Shop Controllers (Meyer, 1990) developed under EP 932, has been adapted, reapplied and considered as a reference model by the EP 2434 consortium.

The project management of this large consortium (Fig. 1) is based upon a four-task structure and, consequently, four task leaders have been appointed with responsibilities for role definition, attainment of objectives, deliverables and overall technical reporting. The project has three application sites: the IC factory of CEA-LETI in Grenoble-France, the electronics factory of Philips Videowerk in Vienna-Austria and Pirelli Tyre Holding with all of its tyre factories in Europe.

This paper will focus on developments, experiences and partners linked with the Pirelli Tyre Holding application site only.

Pirelli Tyre has also been appointed as the Task leader for Task 3 "Integration: Factory implementation projects testing CIM Controllers in small, medium and large batch manufacturing", and about this role we will investigate later.

PARTICIPANTS	COUNTRY	ROLE
Philips GmbH	D	C
BICC Technologies Ltd	UK	P
Fraunhofer Institut - IPA Stuttgart	D	P
FIAR SpA	I	P
Pirelli SpA	I	P
SGN	F	P
CEA	F	P
Tecnicas Reunidas	E	P
Steria	F	P
Alcatel Austria - ELIN	EFTA	A
Philips Videowerk Vienna VIW	EFTA	A
Noratom - Avenir	EFTA	N
FZI Karlsruhe	D	A
RWTH Aachen University	D	A

IFA University Hannover	D	A
ARS SpA	I	A
Politecnico Milano	I	A
Université de Savoie	F	A
TITN	F	A
SISAV Srl	I	A

Fig. 1

As in EP 932, the methodological approach used by the consortium follows the GRAI Method (Doumeingts, 1984), making available a common functional and organisational analysis framework to define a factory reference model. The GRAI Method was developed by G. Doumeingts at the GRAI Laboratoires of the University of Bordeaux. Its objective is to help in designing Production Management Systems (specifically the decisional system) and is based on a Conceptual Model, certain Graphic Tools and a Structured Methodological Approach.

The Conceptual Model provides the basic rules needed to design a manufacturing system, describing the structure of a Production Management System (PMS) in terms of Decision Centers and the details of the activities related to each one. The GRAI GRID and GRAI NETS are the Graphic Tools; the former provides a hierarchical representation of the whole structure of the Decision Centers of a PMS, and the latter gives the structure of the various activities contained within each Decision Center.

The project has produced a generic GRAI GRID model, to define the working area of EP 2434 (Fig. 2). Later, the necessary analysis and activities to customise it according to the real structure and needs of each application site have been performed.

Each Decision Center defined in such a model has been supported by a set of expert systems developed by the partners, called basic CIM Controllers, all with the same architecture (Fig. 3).

Some of those CIM Controllers, which successfully passed the factory test, have been further developed towards marketable products.

A set of CIM Controllers placed at the same hierarchical level is called Workcell or Shop Coordinator (Fig. 4), where horizontal integration between CIM Controllers is described via requests and vertical integration via decision frames, following the GRAI terminology of decision network analysis. Within one hierarchical level, each CIM Controller has a certain level of local autonomy to carry out its own control decisions, even if the decision activities are coordinated by the CIM Controller of the Production Planning function.

Pirelli Tyre followed this approach to system development in EP 2434 since the first requirement phase, as described in the next section.

5. Pirelli Tyre Holding requirements analysis

By using the GRAI methodology to analyse the information flow and the Decision Centers of the standard Pirelli Tyre Operating Unit, the Information and Decision Grids and the Decision Nets of the PTH customised GRAI reference model have been created to support software modules development within EP 2434.

The Information and Decision Grids represent, in terms of informational and decisional flows, the relationships among the various Decision Centers (and thus correspond to those among the CIM Controllers), which together produce the entire decision making structure.

Via the Decision Nets, the activities and decisions of each element in the Grid (or rather, each Decision Center or CIM Controller) have then been described.

FUNCTION HORIZON /PERIOD		SALES MGMT	OPERATIONAL MANAGEMENT			RESOURCE MANAGEMENT			PRODUCT DESIGN
			PRODUC TION	QUALITY	MAINTEN ANCE	MATERIAL	PERSON.	EQUIP.	
COMPANY FACTORY	4 YEARS 1 YEAR		STRATEGIC PLANNING						
FACTORY SHOP	1 YEAR 1 MONTH	FORECASTS	GENERAL MASTER PLAN	DETERMINE & ANALYZE QUALITY PROCEDURES	MASTER PLAN FOR MAINTENANCE	DEFINITION OF VENDOR CAPACITIES, ORDERS FOR COMMON PARTS	PERSONNEL POLICY FORECAST CAPACITIES	EQUIPMENT POLICY DEVELOP NEW EQUIPMENT	DEVELOP NEW PRODUCTS
SHOP	6 WEEKS 1 WEEK	CUSTOMER ORDERS	MASTER PLAN	COLLECT & ANALYZE QUALITY INFORMATION	ANALYSIS & MAINTENANCE PLANNING	SHORT TERM ORDERS EXTERNAL SUBASSYS	PLAN ALLOCATION OF PEOPLE	PLAN INSTALLATION MODIFICATION OF EQUIPMENT	MODIFY EXISTING PRODUCTS
	3 DAYS 1 DAY		RELEASE SHOP ORDERS	DETAILED CHECKING & ANALYSIS OF SAMPLES	PLANNING OF MAINTENANCE ORDERS	RESERVE PARTS & SUBASSYS <small>qty./part/order</small>	ALLOCATION OF PEOPLE TO JIT OR SUBAS.PROD.	RESERVATION OF EQUIPMENT	DECIDE ON ALTERNATIVE SUBASSYS
WORKCELL	3 SHIFTS 1 HOUR		SEQUENCE ORDERS FOR JIT & SUBASSY PRODUCTION	MEASURE & CHECK QUALITY	DIAGNOSIS REPAIRING SERVICING	STORES OUTPUT	ADJUST ALLOCATION	SET UP	DECIDE ON ALTERNATIVE COMPONENTS

Fig. 2 GRAI GRID (WORKING AREA OF THE PROJECT)

Pirelli Tyre has established an international working group (CIM GROUP) composed of both members of the corporate and production unit staffs.

The task of the group has been to identify and agree upon a common functional reference structure for the whole Tyre Sector and then to trace the actual system coverage over this.

To this end, both general and specific GRAI Grids were developed and analysed, covering the complete manufacturing organisation of a PTH standard unit and the three most important functions of Production Planning, Maintenance and Quality.

The final picture of systems coverage was reached thanks to positive collaboration from the end users. Such involvement provided indications about the levels of use, satisfaction and automation related to each system and, thus, completed the results already obtained from the preceding analysis.

The contents of these studies constituted a confidential document, to be considered as a base for future CIM developments in PTH. The part of that document which could be published also provided the guidelines for the development of CIM advanced IT systems within EP 2434 for Pirelli Tyre application site.

Unless one has the fortune to apply a project like EP 2434 to the definition of a new factory, with a vast commitment also to systems, it is unlikely that the resulting products, although integrated and integrable, would be sufficient to manage completely the reality of manufacturing.

Instead, the more common situation is that of applying advanced IT to partially substitute or run in conjunction with traditional systems, already existent and consolidated within the factory.

This point of view must certainly not be considered as a secondary one (as very often happens) and could even give greater challenges.

In fact, the ability to develop advanced IT tools, which work in conjunction with their traditional counterparts, could provide particular advantages: the elaboration of knowledgebased decisions (outside the scope and possibilities of standard systems) could lead to the generation of optimised solutions, which can then be sent back to traditional systems, better reaching the goals, gradually integrating the two environments and reducing the risks.

The number of cases is increasing, particularly in recent years, in which traditional systems fail to give a performance in line with a very rapidly changing production environment and market. However, in a factory, today and for some time to come, there are situations in which the application of advanced IT tools is more a waste than an advantage.

A real plus is achievable through a precise and reasoned analysis of how, to what extent and where to integrate the two levels of technology, reaching the expected performances using tools consistent with the necessary levels of elaboration and decision.

At this point the reference picture was considered to be ready. The following second phase aimed at identifying those areas not yet covered, only partially or unsatisfactorily covered by existing systems and that could be inserted into the frame of EP 2434.

This method of working had the advantage of securing the commitment of all levels (from top management to operative, from corporate to factories) through their involvement in both the analysis and decision phases. Hence avoiding the common crises of rejection by industrial partners which is often associated with research projects.

The role of the Programme Manager inside the application site now becomes vital, as he acts as the interface between three worlds, who must collaborate for the success of the project:

- the top management (in our case this includes both Corporate and Operating Unit levels);
- the middle management (generally in the factories where products are physically developed, tested and installed);
- the consortium partners.

The kinds of approach and motivation linked to these three categories differ in terms of substance, frequency and methods. Thus the utmost care is called for, to avoid a lack of comprehension and interest from anyone of the three, which would consequently jeopardise the work and objectives of the others.

Once convinced, top management should not be forgotten but, instead, be involved in the results as and when they happen, in the solution of problems at high level and in the continuous assurance of middle management commitment.

The consortium partners that, particularly in our case, find themselves working in factories that they do not know, often in different countries, with notable difficulties, should be helped favouring their contacts with factory personnel from development of products to their final installation.

Generally, it is at this point that the contrast between partners and factory personnel (operative middle management who has been previously mentioned) results in disillusion among the expectations of both parties.

The fact is that, in general, it is not possible to enter into a well organised and managed factory with massive interventions (as it has been done in EP 2434 with the development of six products within PTH) and find the level of attention and participation required by partners, without damaging the normal running of production.

As a consequence, this problem provides one of the most important jobs of the Programme Manager of the application site. The Programme Manager must proportion the collaboration of the different factories in line with the actual resources available, taking in careful consideration also workloads and skills required.

It is true that the success of a project depends greatly upon the organisational structure involved, the project's conceptual origins and the professionalism of whoever develops the software. However, having said all of this, without the available and active collaboration of the final users, who are vital for know-how acquisition and testing phases, the end result could be placed in big danger.

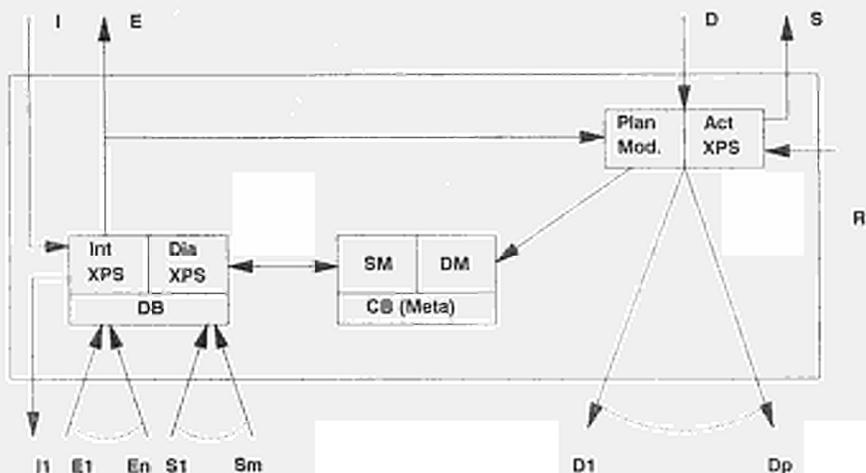
The Task Leader of an application task of an ESPRIT project must be aware of this kind of possible problems and difficulties: for this reason the Programme Manager of Pirelli Tyre Holding application site has been appointed Task Leader of such a task within EP 2434.

In Pirelli Tyre the analysis has been done with much attention given not only to the presence or lack of a certain problem in a certain factory, but also it has taken into account the availability of resources, their pre-disposition and their relative workloads. In consequence a set of subprojects has been created, which distribute partners within the various factories in the Group, while at the same time assuring the greatest possible level of satisfaction for all entities involved.

Sometimes it has been considered more serious to negatively respond to requests for new activities and experimentations (gaining thus the esteem of partners and factory personnel) rather than accept and provide a poor level of service, due to a lack of resources, and certainly not of willingness.

The situation which ensued from these endeavours is summarised in Fig. 5, showing, for every factory of PTH within EP 2434, which partners have been involved (written

above the line) and the names of the products developed there (written below the line). These products shall be described in the following section.



MODELS:

- SM Static Model
DM Dynamic Model

DECISION FLOW:

- d Decision frame to upper level
d1-dp Decision frame to lower level
r Request from/to same level

INFORMATION FLOW:

- e Status of resources to upper level
e1-en Status of resources from lower level
i Status of upper level CIM Controller
i1-im Status of this CIM Controller to lower level
s Decision frame status to upper level
s1-sp Decision frame status from lower level
u Request status from/to same level

Fig. 3 CIM CONTROLLER

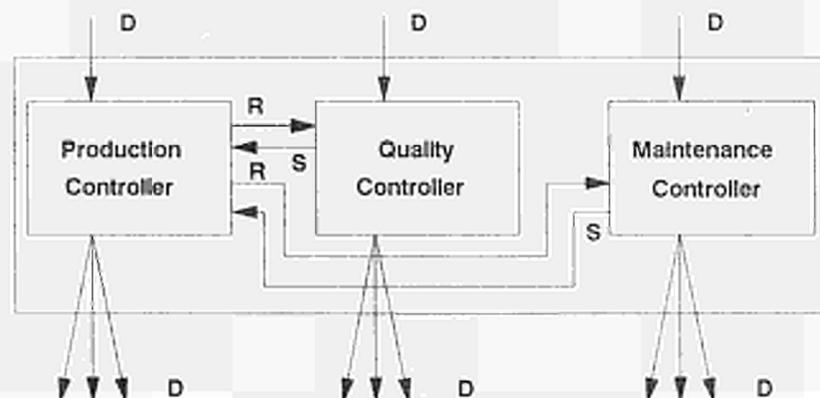


Fig. 4 CIM COORDINATOR

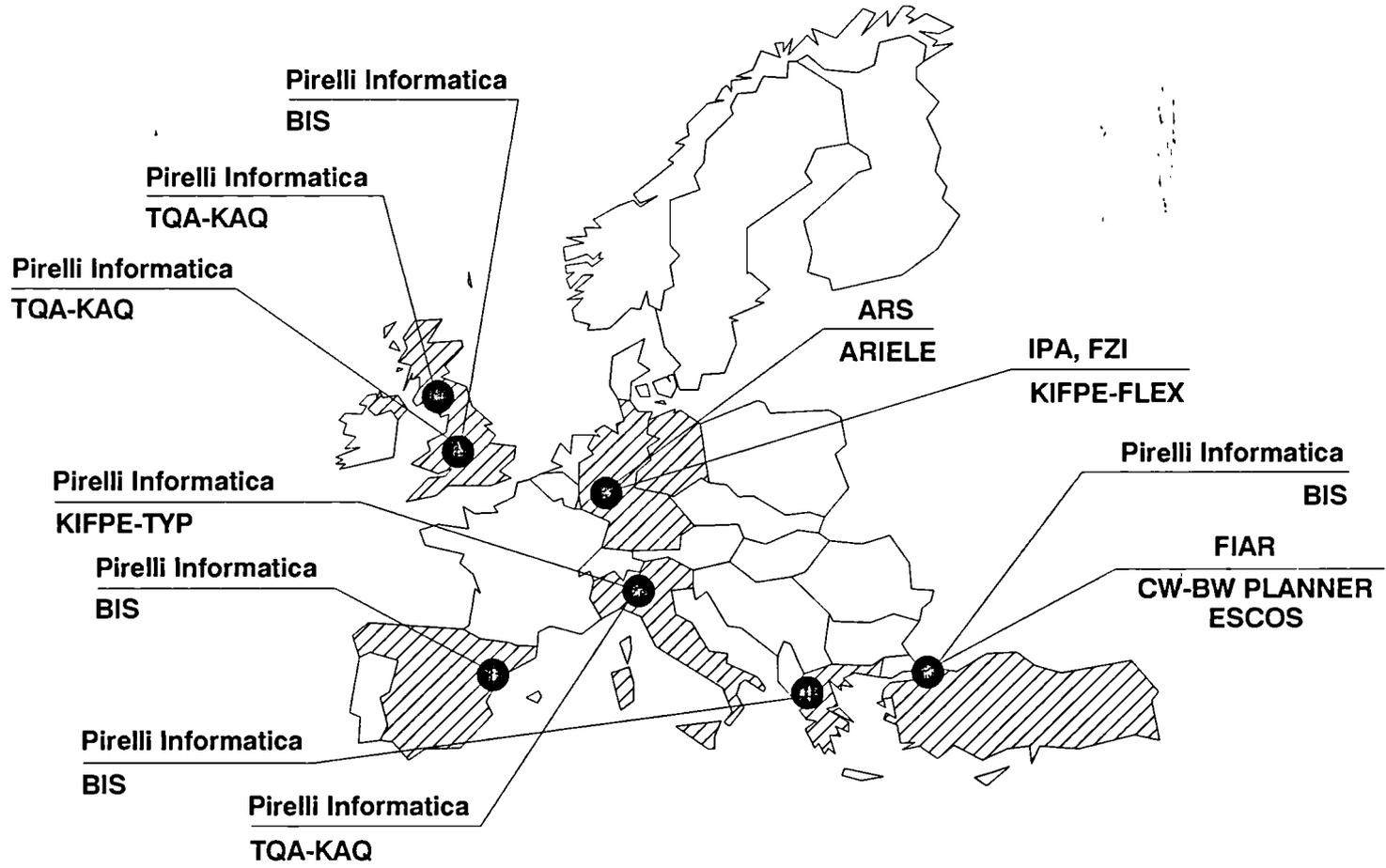


Fig. 5 PRODUCTS DEVELOPED WITHIN PTH APPLICATION SITE

In the case of PTH, some products have been jointly developed from the very beginning (both in the specification and test phases) by various factories in different EC countries. Such collaboration has been achieved through special international working groups, which have allowed a deeper level of analysis and easier simultaneous installations (with all the advantages of different user feedbacks and operating conditions on the same product).

In the cases in which this joint cross-group involvement has not been thought necessary or viable, due to reasons already mentioned, every effort has been made to keep the relative specifications as general as possible.

Thus the door has been left open for future extensions to products, once tested, installed and consolidated in a single factory.

However, in both cases, the future applicability of each product in all PTH factories has been considered a pre-requisite.

6. Products developed within PTH application site

The ESPRIT 2434 project has led to the creation of many products, each of which has been customised to one out of the three application sites (CEA-LETI, Pirelli Tyre Holding and Philips VideowerK Vienna).

In particular, the Pirelli Tyre Holding application site corresponds to the development of 6 such products, which will be described in the following section, involving all the factories shown in the map in Fig. 5.

The first product (BIS) under examination shall be looked at in detail to illustrate the basic formula which was used during all product developments.

6.1 BIS - Banbury Information System

BIS (Guida & Basaglia, 1990a; 1990b; 1991a; 1991b), developed in close collaboration with Pirelli Informatica, solves the complex problem of scheduling the compound production area of a tyre factory.

The function of the area lies at the very beginning of the production process and consequently represents the first and very important link in the internal supply chain.

Such an area (5 to 9 machines) processes about 50 jobs per day, which correspond on average to 150 machine operations (since each job is produced in about 3 operations).

Schedules were generated and revised by hand on a daily basis (taking about 4 hours) with an horizon of 4 shifts.

Flexibility has been built into the BIS system to allow its easy transference to all Pirelli Tyre factories having a compound area.

In order to achieve full flexibility a lot of emphasis was placed on the requirements definition phase of the system. In particular, users from different factories were directly involved in the project from the very beginning. Through the collaboration of end users (factory level), corporate staff and IT experts within an International Pirelli Working Group, a common set of requirements was established.

Continuous user involvement characterised the development of the product, from the requirements phase onwards. For example, the testing and subsequent feedback on each prototype (two, in this case) was carried out within the various factories by the users and then the results analysed by the Working Group in common sessions.

In the generic GRAI GRID of the project, BIS covers the functions of Production (Operational Management) and Material (Resource Management) from the Workcell level ($f = 1$ hour; $h = 3$ shifts) to the Shop level ($f = 1$ week; $h = 6$ weeks).

BIS is a Knowledge Based system which uses the blackboard architectural model to integrate all techniques suitable to the solution of scheduling problems, such as Operational Research, Artificial Intelligence and conventional programming. The product has been developed on a 386-based hardware platform under an MS/DOS operating system (using Paradox as relational database, OPS83 for the scheduling architecture and knowledge-based modules, C Language for the integration of database and scheduling modules, FSCREEN from Pirelli Informatica as user interface).

The target system has been delivered in September 1991 to four Pirelli Tyre factories within the following countries: Spain, Turkey, England and Greece. Further system applications are planned to cover the remaining tyre factories producing compounds, starting from 1992.

Industrial benefits

The flexible and modular architecture has led to the creation of a system which is completely identical for all application sites, and this will permit fast and effective installations in addition to easier system maintenance.

BIS can take into account all possible relevant parameters (always with the same level of priority previously agreed) when calculating a schedule, something which is almost impossible to do by hand: this guarantees a more constant level of quality in the schedules produced in different times and different factories.

The efficiency of the system has dramatically reduced the time needed by the user: from the four hours per day mentioned before, the planner is now able to better perform the same task in maximum one hour per day, where the schedule generation is only taking about 2 minutes and the rest is data preparation and checking. This already excellent result is expected to improve again when all the link of BIS with the existing systems will be fully automatic and no more (by exception only) data preparation and checking will be required.

This fully automatic integration with existing factory information systems will also improve the capability to respond in a very short time to all possible factory disturbances, which is becoming vital in relation with the increased level of flexibility required.

Stock levels are more constant and balanced with regard to total mix, total quantity and the distribution between final compounds and intermediate stages.

Machine usage has increased due to reduced set-up times and more balanced saturation levels.

The system increases the percentage of usage of high priority machines, which in turn increases productivity and product quality (from about 50 % up to a more constant level of about 90 % in normal situation and about 75-80 % during emergencies)

The future efforts will be concentrated on improving the on-line scheduling and rescheduling capabilities of BIS, to allow for more frequent use of the system by the production supervisor on shifts when the compound area planner is not available. This result will speed up once more the reaction of this area in case of possible deviations from the planned production in any other one.

6.2 TQA - Tyre Quality Assistant

TQA (Guida & Majocchi, 1989; Guida & Basaglia, 1991c; Basaglia & al., 1992), developed in close collaboration with Pirelli Informatica, is a knowledge-based decision support system which assists the manufacturing managers at the operative level of a tyre factory in choosing the best solutions to complex quality related problems as and when they arise. Such decision making is based upon specialised knowledge and individual work experience, both of which are difficult to provide for full 24 hours factory coverage, hence the need for a support system.

TQA is fully integrated with the already existing conventional system of Final Inspection (present in all Pirelli Tyre factories around the world) and gets information from this system about the defects detected in the quality area as soon as they are declared by the workers.

Since Final Inspection is an on-line system running 24 hours a day, TQA is able to follow and monitor the actual situation about quality in the shop floor and to give supports to decision makers.

In the generic GRAI GRID of the project, TQA covers at the Workcell level (f = 1 hour; h = 3 shifts) the functions of Maintenance and Quality (Operational Management) and at the Shop level (f = 1 week; h = 6 weeks) the function of Quality (Operational Management).

TQA uses a knowledge base to represent the expertise of the shop floor personnel. The target system was originally geared to a DEC Vax Station under MicroVMS operating system, (using Nexpert Object as KB development tool, VAX Windowing System as user interface, C Language and KERMIT (MIT) for file transfer with existing systems).

The relatively high cost for the hardware platform in comparison with a very difficult rigorous cost benefit evaluation (since the benefits are much more qualitative than quantitative), suggested to revise the system to run on IBM PS/2 under MS WINDOWS. In particular, it was noted from the first installation that users need to be able to maintain the system themselves without the intervention of an IT specialist. From this need the development of the third product (KAQ-Knowledge Acquisition for Quality) was originated (see description below).

Target System on the first hardware configuration was delivered to a Pirelli Tyre factory in Torino, Italy in 1990.

System update has now been completed and the new version is available since March 1991. Further installation are planned in two factories in Pirelli Ltd UK from end 1991. Other installations will follow, according to the available resources.

Industrial benefits

The efficiency and effectiveness of decision making has clearly increased, (the ratio between problems successfully solved and total problems declared has risen) and the time taken to respond to identified problems has been considerably lowered.

The level of skill required to perform such specialised diagnosis has been reduced and quality related decisions could now be taken by the production supervisor without calling the quality assistant when he is not in the shop floor.

The process to have "quality" directly made by production people, without the need of a skilled "quality man", is very much favoured by the use of the system. The training process on quality related problems is sped up and it doesn't need the constant pre-sence of the tutor anymore.

6.3 KAQ - Knowledge Acquisition for Quality

KAQ (Guida & Basaglia, 1991c), as TQA developed in close collaboration with Pirelli Informatica, is a knowledge acquisition tool to be used by the end users of TQA. It is the medium by which knowledge is transferred to and updated within TQA, thus KAQ is used to create and then maintain the TQA Knowledge Base.

The system was developed using the same criteria as TQA. Both the hardware and software environments are the same as those used in the updated version of TQA (eg. Nexpert Object and C programming language on a IBM PS/2 with MS-WINDOWS).

Final version has been delivered with TQA update in March 1991 and installations are following the TQA planning.

Industrial benefits

This tool gives and encourages complete ownership of the TQA system to the user. In addition to being self sufficient with respect to running and maintaining the system, the user, by nature of his front line position, can detect and react quicker to knowledge-base discrepancies rather than IT specialists.

Errors of data content, due to poor communications between factory experts and IT experts, are now avoided by the direct access of data via KAQ.

Since the IT experts are based in Milan at company headquarters and TQA will be installed in all the Pirelli Tyre factories around the world (and only a very low level of interaction is practically allowable), KAQ will in fact make the installations feasible.

6.4 ARIELE

Ariele (Basaglia & al., 1992), developed in close collaboration with ARS SpA - Milano, is an high level production planning system which optimises global costs while integrating the objectives and constraints of Sales and Resource Management. The scope of the system can span different production strategies and time horizons (ranging from one week to 16 months).

Simulation functions are a very important part, since the real use of the system is based on two fundamental steps:

- unconstrained run, aimed at demonstrating the actual difference between the requests from Sales and the capacity of the factory (different runs will allow to take decisions about capacity, stocks and costs via the use of several parameters available);
- knowledge-based balancing run, aimed at matching requirements, stocks and capacity in order to achieve a feasible plan, always indicating the deviation from the unconstrained, ideal and cost optimum one.

In the generic GRAI GRID of the project, ARIELE covers the functions of Sales Management, Production (Operational Management), Material, Personnel and Equipment (Resource Management) from the Shop level (f=1 week; h=6 weeks) to the Factory/Shop level (f=1 month; h=1 year).

Ariele is a Knowledge Based system developed on a 386-based hardware platform under an MS/DOS operating system (using DB3 as relational database, Golden Common Lisp for knowledge-based and inference engine modules and C Language for the graphical representation).

The last prototype of the system has already been used (December 1990) in the German Pirelli Tyre factory with final users and real data during 1991 Management Plan preparation, giving excellent results and response times. Final testing will be completed by October 1991 and installation in the same factory is planned for December 1991, in order to be used for 1992 Management Plan.

Industrial benefits

This high level planning has proved to be very useful during the last four years, when the predecessor of this system has been used in Pirelli Germany (it was a traditional system, mainly based on what-if functions of Lotus 123).

During this time a lot of changes and enhancements (directly suggested by the final users) have been made and constitute now the basis for the requirements of the new system. The need of an expert system appeared very clearly when the users started to really use the system and stress its possibilities.

An high level planning tool already demonstrated to be very useful to simulate the operative environment during management plan phases, in order to fully understand the gap between the reality and the cost optimum situation, to take fact-based decisions rather than feeling-based ones.

The output given by Ariele after the knowledge-based balancing run is taken by the operative systems to manage it monthly, weekly and daily.

6.5 KIFPE - Key Indicators for Factory Performance Evaluation

KIFPE (Guida & Basaglia, 1991d) is a set of knowledge based decision support systems aimed at supplying manufacturing decision makers with an appropriate set of key performance indicators.

Further to this, it aims at supporting the computation, evaluation, interpretation and diagnosis of the said key indicators.

The array of factors which are related to manufacturing decision making is so vast that the system development has been split into modules. Each module is concerned with one critical performance indicator which may lie in any organisational area.

Two modules have been developed to date. The first concerns a man-productivity indicator called "TYP" (production related), carried out at Milan Corporate headquarters of Tyre Sector. The second deals with the analysis of a flexibility indicator (market related), and it is still in the first prototype phase at the German Pirelli Tyre plant.

In the generic GRAI GRID of the project, KIFPE covers from the Shop level (f = 1 day; h = 3 days) to the Factory/Shop level (f = 1 month; h = 1 year) the functions of Personnel and Equipment (Resource Management) and to the Shop level (f = 1 week; h = 6 weeks) the function of Material (Resource Management).

KIFPE-TYP has been developed (in close collaboration with Pirelli Informatica) on a 386 based PC under MS-DOS operating system, using a mix of software technologies (Lotus for the input of rough data, QuattroPro for support and generation of graphics and Guru for the development of the knowledge-based component of the system).

This and all the future modules will be linked together via a network at corporate headquarters in Milan and will receive the required information directly from the Operative Units via electronic data transfer with frequencies and horizons defined in the organisational analysis.

The hand over of first KIFPE module to final users at the Pirelli Tyre Sector headquarters in Milan has been completed in May 1991.

The prototype phase for the flexibility related indicator (developed in close collaboration with IPA Stuttgart and FZI Karlsruhe) is expected to be concluded by end 1991. Analysis and definition of new indicators needed are presently under way at corporate level, with a strong cooperation of users from operative level (factories) via international working groups; this phase will allow to start, later January 1992, the development phase of the remaining modules and their integration.

Industrial benefits ..

The system KIFPE-TYP can provide the user with characteristics and properties related to indicators under analysis as opposed to just qualitative data. An explanation of the obtained results can be generated in natural language.

The computational power provided by KIFPE-TYP has dramatically reduced the manpower needed to prepare graphics and reports which are virtually produced automatically. The above factors help the users to generate very quickly the right set of information in order to consistently support the correct decisions at their level and at the higher levels, when necessary.

These two modules are only a very partial view of a complex architecture aimed at helping the decision making activities at all level of the chain.

Any singular module will help the user directly involved with the control of a particular indicator, but at the same time the different modules linked together will allow higher level of analysis and decision making by the managers of different areas (production, quality,) at corporate level.

The same models used to control and analyse the indicators at corporate level will be used at factory level by factory managers. This will create an agreed set of indicators and rules to be used by all decision levels to analyse facts and react accordingly, having all the needed information available.

The philosophy of Tableau de Bord (Saulou, 1984) will be brought into the project and all the modules will be developed accordingly.

These modules will be linked together according to the needs of the defined decision making structure, and the knowledge-based part of them will be developed in order to take the best advantage from their collaboration.

6.6 CW - BW PLANNER & ESCOS

CW and BW Planner (Basaglia & al., 1992) are production planning expert systems able to assist the planner of these two workcells in allocating jobs, machines and resources.

These steps of the production process are the last two, right before the Final Quality Check and allow the semifinished product to finally get the shape of a tyre we all know. Particular attention has been paid to consider the impact of Final Quality Control, Equipment Maintenance and Resource Management constraints on this planning process.

ESCOS (Basaglia & al., 1992) is a data interpretation and management software to allow coordination of activities and cooperation at shop floor level of expert systems (or more traditional software tools) developed at workcell level, via a blackboard-based communication paradigm.

Two main activities require the ESCOS intervention: to maintain the consistency of data files (orders, policies, word models, constraints and schedules stored in the black-

board) and to solve the shop floor management conflicts as soon as a perturbation impacts the production plan.

In the generic GRAI GRID of the project, CBW PLANNER & ESCOS cover at the Workcell level ($f=1$ hour; $h=3$ shifts) the function of Personnel (Resource Management) and at the Shop level ($f=1$ day; $h=3$ days) the functions of Production (Operational Management) and Material (Resource Management).

CW - BW Planner and ESCOS have been developed (in close collaboration with FIAR SpA - Milano) on a SUN workstation under Unix Operating System, using C and OPS83, for rule-based modules, as implementation languages.

The target system will be delivered to Pirelli Tyre Corporate at end 1991 to be finally tested on two factories in Italy and Turkey with end users and real operative data. The main use of the systems during their test phase has been as simulation tools for generation and analysis of what-if scenarios at workcell level for a new factory in Turkey.

Industrial benefits

A considerable improvement in the quality of the schedules generated (machine allocation level, number of jobs allocated,...) as been achieved, taking into account PQM interactions and also minimising the time required for the completion of the task (from about 4 hours to few minutes for 1 week plan).

These two systems represent a valid opportunity to move from the traditional way of programming different step of a tyre factory following a predetermined sequence. Planning them together (or at least having them strictly linked) will allow to save time and to be much more effective, being able to immediately and interactively revise one step as soon as the other one shows troubles.

The reaction to production, quality or maintenance related disturbances is sped up by this interaction and by the ESCOS software. ESCOS collects and analyses information and possible actions from the specialised functions of the factory, to solve these problems coordinating them towards the right solution.

What-if functions at shop floor level are very important not only from a very practical point of view on daily planning, but also as simulation tools during the design of a new factory, allowing to simulate different situations and to design the machines lay-out, the coupling size-machine and the maintenance procedures accordingly.

6.7 Global view of effort within PTH application site

In the ESPRIT 932 and 2434 Pirelli focused its development efforts on 3 workcells (namely, Banbury, Building and Curing) and, with reference to the GRAI GRID shown in Fig.2 on the Workcell, Shop and Factory levels. According to the reference architecture of the CIM Controller and CIM Coordinator, 15 different CIM Controllers would have been required, in order to completely cover the areas of interest.

Out of these 15 CIM Controllers, only a few have been actually developed within ESPRIT 2434, as described in the previous sections (their final coverage of the project GRAI Grid is shown in Fig.6).

In fact, some of these areas have been already covered in ESPRIT 932. Others, as resulted after the work of the Pirelli CIM GROUP, are already well controlled by existing systems and others do not require expert systems technology to be solved, or the expert systems component is very small with respect to the whole system. Finally, others required development resources far larger than the ones available within the EP 2434 project.

The CIM controllers developed, all comply with the reference architecture shown in Fig. 3, even if some of them do not include all the components.

For instance, the Quality controller TQA includes the Diagnosis and Interpretation modules, as well as the Static and Dynamic databases.

On the other side, it does not include a complete Action Planning component, since it is not needed, according to the requirements coming from the shop floor of the Pirelli application site.

However, thanks to the modular architecture of the CIM Controller, missing components can be easily added, if the functional analysis of Pirelli Tyre factories will show their need. This is a typical case, in which the concepts promoted by EP 2434 revealed to be flexible enough to be easily adapted to individual partners exigencies.

Another good example is the possibility to use the same CIM Controller architecture to develop systems laying across different levels and/or functions, to better match organisational issues of specific application sites.

For instance, the BIS system, while consistent with the CIM Controller general architecture, covers the Production and Material functions, from the Workcell to the Shop levels.

7. Conclusions

Large industrial groups in which IT is a service rather than an end-product have a need for the type of approach described, in order to convince all levels of management to allow the experimentation of non-advanced systems within their factories.

In either case this commitment will only be granted if the location, function and utility of the final product is very clearly outlined.

In practice, the same type of direct and continuous involvement, which now characterises the role of end users to guarantee a comprehensive development and usage of any system, should also be applied to all levels of management. When this step is not carefully handled, cases occur, not infrequently, in which application sites blandly participate in the projects and this leads to a virtual "dragging" of such sites through the motions.

Consequently, the partners are left without motivation because their work lacks stability and concreteness and, in general, they do not have the satisfaction of being able to positively test and install their products in factories.

This is thought to be one of the most important tasks that an application site Programme Manager must carry out within his company, to assure the integrity of both the application site involvement and the collaboration of the other partners.

While the advantages of the participation of Research Centers, Universities and IT Companies in ESPRIT projects are evident and easily found, there is still the impression that companies playing the role of application site do not gain all of the potential offered by this Programme.

Unfortunately, there are still too many cases in which IT companies look for industrial partners in the hope of involving their application sites in these projects.

The best opportunities only spring from collaborations which from the very beginning have their roots in concrete problems.

The experience of Pirelli Tyre Holding has shown that the opportunity offered, when exploited with integrity and willingness, is enormous.

A large number of partners with different experience, know-how and skills (the scope of which would be practically impossible to obtain within one company) are available through the consortia.

FUNCTION HORIZON /PERIOD		SALES MGMT	OPERATIONAL MANAGEMENT			RESOURCE MANAGEMENT			PRODUCT DESIGN
			PRODUC TION	QUALITY	MAINTENANCE	MATERIAL	PERSON.	EQUIP.	
COMPANY FACTORY	4 YEARS 1 YEAR		STRATEGIC PLANNING						
FACTORY SHOP	1 YEAR 1 MONTH	FORECASTS	GENERAL MASTER PLAN	DETERMINE & ANALYZE QUALITY PROCEDURES	MASTER PLAN FOR MAINTENANCE	DEFINITION OF VENDOR CAPACITIES, ORDERS FOR COMMON PARTS	PERSONNEL POLICY FORECAST CAPACITIES	EQUIPMENT POLICY DEVELOP NEW EQUIPMENT	DEVELOP NEW PRODUCTS
SHOP	6 WEEKS 1 WEEK	CUSTOMER ORDERS	MASTER PLAN	COLLECT & ANALYZE QUALITY INFORMATION	ANALYSIS & MAINTENANCE PLANNING	SHORT TERM ORDERS EXTERNAL SUBASSYS	PLAN ALLOCATION OF PEOPLE	PLAN INSTALLATION MODIFICATION OF EQUIPMENT	MODIFY EXISTING PRODUCTS
	3 DAYS 1 DAY		RELEASE SHOP ORDERS	DETAILED CHECKING & ANALYSIS OF SAMPLES	PLANNING OF MAINTENANCE ORDERS	RESERVE PARTS & SUBASSYS <i>qty./part/order</i>	ALLOCATION OF PEOPLE TO JIT OR SUBAS.PROD	RESERVATION OF EQUIPMENT	DECIDE ON ALTERNATIVE SUBASSYS
WORKCELL	3 SHIFTS 1 HOUR		SEQUENCE ORDERS FOR JIT & SUBASSY PRODUCTION	MEASURE & CHECK QUALITY	DIAGNOSIS REPAIRING SERVICING	STORES OUTPUT	ADJUST ALLOCATION	SET UP	DECIDE ON ALTERNATIVE COMPONENTS

Fig. 6 GRAI GRID COVERED BY THE PRODUCTS DEVELOPED WITHIN PTH APPLICATION SITE

Other than the ability to clarify and follow objectives and work well together, little less is required.

It seems, however, that winning the commitment of the management of an industrial group is much more complex than achieving that of an IT company.

The fact remains that results are still the best medium through which to achieve commitment, but they are missing at the beginning of any new endeavour, if not acquired from preceding projects.

The catalytic role played by results (in the preparation phases of the project and when searching for consortium partners) could be sometimes substituted by a clear methodological approach.

Such a formula must be treated with the utmost seriousness and be reciprocal between all partners who, in turn must guarantee their total commitment from the beginning, in order to achieve the project objectives.

However, it is vital to guarantee commitment from the factories that the attention of the application site is mainly focused on areas identified as important, and not on themes of exclusive interest of IT companies.

Furthermore, the greatest advantages can be gained if such areas really require the kind of international exchange of experience and know-how that ESPRIT could provide.

The importance of a Programme such as ESPRIT should not be limited only to its financial contribution (important but not fundamental), but rather it should focus on the exploitation of opportunities of collaboration, which are otherwise generally difficult to obtain.

Once this approach has been understood and put into practice (essentially during the transition from EP 932 to EP 2434), the experience of Pirelli Tyre in ESPRIT was marked by quick and secure progress.

Thus corporate management, unit management, final users of installed systems and, not least, the project partners were all rewarded with a high level of satisfaction.

The door to new projects and further international collaborations is now open, and commitment remains securely rooted in the methodology and the involvement noted so far, not to mention the excellent products already installed in some of the factories and currently being extended to the rest of the Group.

8. References

Basaglia, G., Guida, M., Gusmeroli, S., and Squellati, G., 1992. Tyre factory supervision at shop floor level through integration of expert systems in production scheduling, quality control, To appear in Proc. of 1st Int. Conf. FAIST '92 (Fielded Applications of Intelligent Software Technologies), Toulouse, France.

Doumeings, G. 1984. Methodology to Design CIM and Control of Manufacturing Units, in Methods and Tools for Computer Integrated Manufacturing, U.Rembold & U.Dillmann (eds.), Springer: Berlin

Guida, M., Majocchi, L., 1989. TQA: Un sistema intelligente di supporto alle decisioni per l'operatore della qualita'. AI*IA - National Association for Artificial Intelligence Bulletin.

Guida, M. and Basaglia, G., 1990a. The B.I.S. Project: an experience in integrating Operational Research and Artificial Intelligence in a distributed approach to dynamic scheduling. Proc. of 1990 ESPRIT Conference, Kluwer Academic Publisher, Brussels.

Guida, M. and Basaglia, G., 1990b. Integrating Operational Research and Artificial Intelligence in a distributed approach to dynamic scheduling: the B.I.S. Project. Proc. of 1990 Workshop "Implementing CIM", Saarbrucken, West Germany.

Guida, M. and Basaglia, G., 1991a. The integration of operational research and artificial intelligence as a key factor in building flexible and powerful scheduling systems: the BIS experience. Proc. of Avignon 91 Conf. on Expert Systems and their applications, Avignon, France.

Guida, M. and Basaglia, G., 1991b. Operational research and artificial intelligence in the development of real-time scheduling systems: the BIS project. IJCAI '91 workshop on "advances in interfacing production systems with the real world". Sidney, Australia.

Guida, M. and Basaglia, G., 1991c. An industrial experience in developing knowledge acquisition tools: the TQA & KAQ project. Submitted to Knowledge Acquisition, Academic Press.

Guida, M., and Basaglia, G. 1991d. Supporting manufacturing managers: a knowledge based decision support system approach. Submitted to Decision and Information Technologies, Elsevier Science Publ., North Holland.

Meyer, W., and Walters, H.M.J. 1990. EP 932 Final results: CIM/AI Value Analysis. Proc. of 1990 ESPRIT Conference, Kluwer Academic Publisher, Brussels.

Meyer, W., 1990. Expert Systems in Factory Management - Knowledge-Based CIM. Ellis Horwood Limited, Chichester, England.

Saulou, J.Y., 1984. Il Tableau de Bord del Dirigente. Franco Angeli, Milano, Italia

OMI

Open Microprocessor systems Initiative

Synopsis:

This paper summarises the background behind the Open Microprocessor systems Initiative - the concerns about the European IT industry, and the overall strategy. It is based on the proposal made to the European Commission which led to the creation of the work programme incorporated in the first ESPRIT 3 call.

The second part of the paper details the work going on in a small pilot project - the Microprocessor Architecture Project, intended to explore the areas in more detail so that the OMI initiative did not suffer a gap of two years between the conception and strategy and actual projects starting. OMI/MAP has continued to track and work in many of the key areas since the start of 1991.

1. Historical Background

This initiative has undergone a number of considerable refinement and development changes since its origination.

The initiative has its origins in an work taken by the major European IT companies in response to growing concern of the dependence of Europe on foreign technology. A working group under the chairmanship of Bull examined marketplace needs, various possible approaches, and studied key technological developments. The two companies owning European microprocessor technology - Olivetti (through Acorn; now Advanced RISC Machines, a spin-off from Acorn) and SGS-THOMSON (through INMOS) proposed the current approach, which is to combine European capabilities in systems, software and applications with a new generation of open microprocessor technology, aiming at meeting European requirements from about 1994 onwards and enabling European IT to take a leadership role on the world stage.

This strategy was described in a workshop during the Esprit Conference in November 1989, following which a series of workshops were held to identify more closely the market requirements, refine the requirements for systems development tools, and to detail the proposed projects. At this stage, it was known as EMI - the European Microprocessor Initiative.

Further discussion suggested that this conveyed the "Fortress Europe", which was not the intention, and so it became the OMI - Open Microprocessor Initiative. It was then felt that this title gave the false impression that the initiative was only concerned with microprocessor development, whereas the initiative provided a coherent systems plan, to provide vertical integration from the semiconductor suppliers through to the end users, and so the final title became "Open Microprocessor systems Initiative", to more fully represent this coherent strategy.

The overall aim is for Europe to achieve self-sufficiency in microprocessor technology and its application by 1995. As well as supporting European IT, this will provide a position of strength for international collaboration. Historically, the European semiconductor industry is more coherent in 1991 than it was in 1981; it necessary to build upon this by bringing users and suppliers closer together to reduce time to market and thus

build a competitive edge, which is the factor that will win Europe back a larger part of its home market and increase its worldwide market to a fairer share.

2. Strategy

The major embedded systems markets (automotive, telecoms, office automation, consumer, industrial, aerospace) are expected to grow rapidly over the 1990s. Many of the products in these markets already employ state-of-the-art microprocessor technology. For instance, pollution monitoring and control in industry and in cars, and the efficient use of energy in industrial processes, cars and buildings, depends on embedded microprocessor technology; the number of microprocessors used in these areas will grow rapidly as a result of current European legislation.

In the computing and workstation areas, Europe already has a world-class leadership position in parallel processing and in low cost workstation component technology. However, the competitive situation in both these areas necessitates coordinated efforts on a European scale in order to maintain this position. For the high end workstation market, new microprocessor architectures based on VLIW or dynamic instruction scheduling are under development. **It is not sensible to attempt to develop another conventional RISC processor architecture for the workstation market.** The next major opportunity will come after RISC. It is anticipated that new workstation architectures based on multiprocessing will start to appear in about 1992-1994. The need to move to multiprocessing architectures will create an ideal opportunity to enter the workstation market.

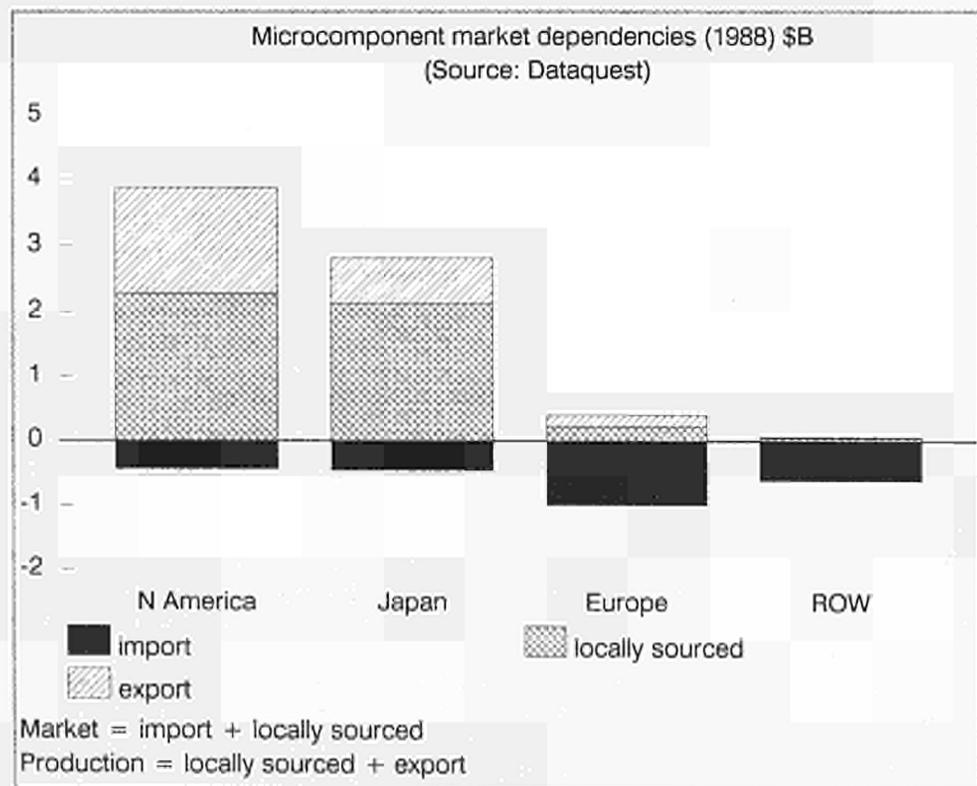


Figure 1 Microcomponent Market Dependencies

The marketplace worldwide for microcomponents is expected to grow at 17% CAGR between 1989 and 1994 (Dataquest). However, the European microcomponent supply industry is growing at only *half* this rate. In 1988, European imports of microcomponents totalled \$992M (representing 82% of the European market). Without a viable European capability, the dependence of the major computing, engineering, telecoms, IT, consumer and aerospace suppliers on non-European microprocessor technology will increase rapidly over the 1990s, reaching 90% of European consumption by 1994. Figure 1 shows how much more European industry is exposed than that of North America and Japan.

The European Electronic Equipment industry also lags behind the rest of the world in its usage of microcomponents (this could well be a consequence of the excessive dependence on imports noted above). Competitiveness in the marketplace depends crucially on successful exploitation of microprocessor technology. Figure 2 shows the percentage microcomputer content of electronic equipment for the major geographic areas.

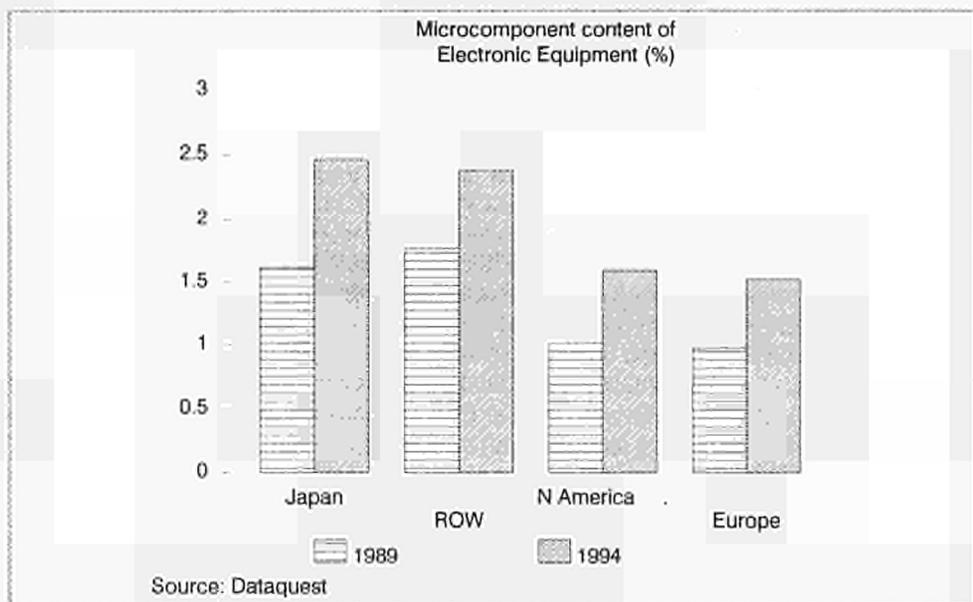


Figure 2 Microcomponent Content of Electronic Equipment

Whilst a growth of 40%-50% is anticipated in this figure over the next five years, this will only result in the European content in 1994 nearly equalling the Japanese content in 1989! Close relationships between suppliers and consumers of microcomponents are needed to accelerate the uptake.

As the 1990's progress, it will be necessary for the applications companies to be able to exploit ever greater levels of component integration in order to retain a competitive position. Use of VLSI systems combining embedded microprocessors, embedded software and applications specific subsystems will increase rapidly. This, in turn, requires much closer relationships between component developers, systems software houses and application developers. (In Japan, this is achieved within large 'vertically integrated' companies.) It is therefore essential for European applications developers to be able to greatly reduce their dependence on non-European sources. It is essential for an Initiative to help forge these relationships within Europe.

The European systems industries have a number of other requirements from the components industries. In particular, an Initiative is needed to:-

- *create leadership opportunities.* Leading edge technology will be exploited in leading edge products.
- *accelerate time to market.* A direct benefit of close relationships with local suppliers.
- *allow product differentiation.* This follows from the the emphasis on mastery of new levels of component integration, the exploitation of procesor macrocells, coupled with the essential close relationships between the systems house and the component and software suppliers.
- *create a position of strength for international agreements.*
- *enlarge international market share.*
- *help correct trade imbalances.*
- *form the basis for standardisation.*
- *accelerate the introduction of formal specification and design techniques.* These will become more and more important to cope with the complexity of digital systems.
- *optimise the use of design manpower.* This follows from the adoption of standards and infrastructure associated with the initiative.

The Open Microprocessor systems Initiative is a programme to provide a complete European microprocessor capability over a 5 year period. The OMI will provide the infrastructure to support the development of new microprocessors and applications based on them, by establishing close links between applications companies, software developers and technology suppliers. The OMI will include the definition of VLSI components and macrocells, but **most of the activity is concerned with the associated system development tools, system software and applications demonstrators.**

The goal of the OMI is to provide state-of-the-art general purpose computing capability. This is essential to the embedded systems of the late 1990s, which will combine general purpose multiprocessors with specialised subsystems on VLSI chips with over 100 million transistors (5cm^2 , 0.3 micron). It also provides a new opportunity for European companies to produce high-performance general purpose computers and workstations.

Several companies have licenced non-European architectures, such as 88000, SPARC and MIPS, and the older CISC architectures. It is notable, however, that licences have not been made available for the market leaders (Intel 80x86, Motorola 680xx).

Existing European technology addresses the embedded systems market and the low-end workstation market; the Inmos transputer providing scalable parallel processing systems and the ARM providing support for conventional general-purpose sequential processing.

There is no European microprocessor architecture which supports high-performance general purpose computing. **The OMI will not attempt to compete with existing RISC chips (88000, SPARC, MIPS). Instead, it will develop the necessary capabilities for the generation after RISC.** These will exploit ultra-fast context switching and dynamic instruction scheduling to provide virtual processing, virtual memory and virtual communications in general purpose parallel computers scalable up to thousands of processors. The OMI will provide a clear migration path from current architectures to the new generation.

The development of software tools and applications packages within the European systems industry forms an essential component of this activity. The OMI provides an

opportunity for European software developers to gain early access to the architectural details of the OMI processors. In the US, such access has been a key factor in creating the US lead in software packages. The OMI will ensure early availability of a development environment to software product developers, and thereby stimulate the European software industry.

The resulting capability will give Europe the critical mass required to compete effectively with the US and Japan, providing the European IT industry with a viable alternative to the use of US or Japanese microprocessor technology. Equally importantly, it will provide Europe with the strength needed to collaborate on an even-handed basis with the US and Japan. The semiconductor components will be widely available (from several European manufacturers) and the associated support in systems development and systems software will be available within the European systems industry. **Existing standards, such as UNIX, will be supported and the open nature of the Initiative will accelerate the establishment of appropriate new standards.** Maximum use will be made of the technical strengths already available in individual companies and in European research activities.

3. The market

The world market for microcomponents (microprocessors, microcontrollers and peripherals) is estimated at \$8081M for 1989, rising at an average CAGR of 17% to \$17526 by 1994. Microprocessors take about 20% of this market, whilst microcontrollers and peripherals both take about 40% (Source: Dataquest).

The market is split geographically as shown in figure 3:-

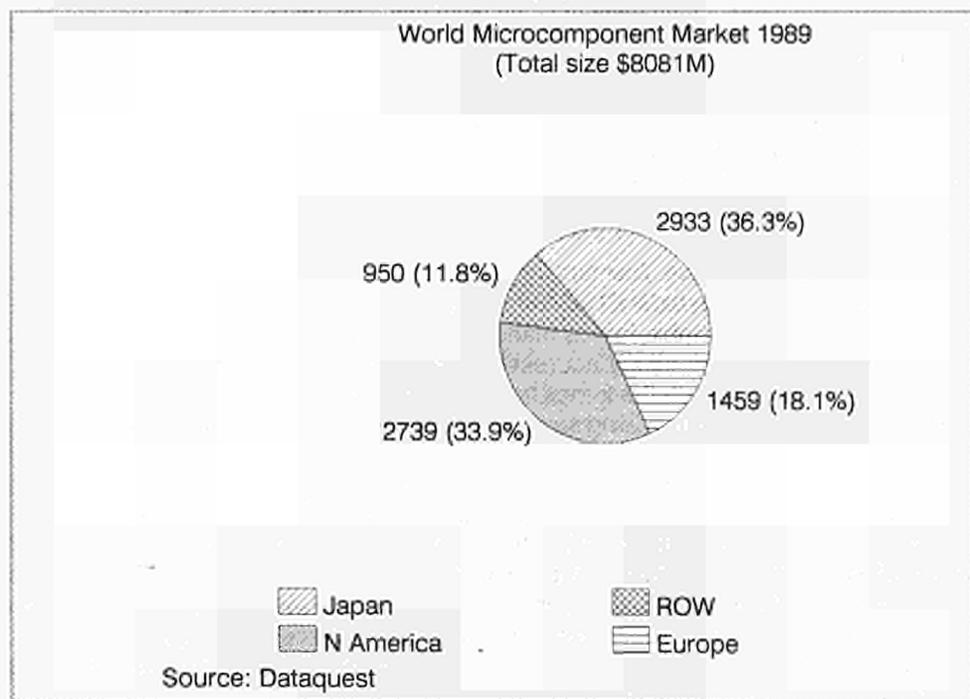


Figure 3 World Microcomponent Market 1989

Although the markets have doubled in size, the geographical split by 1994 is little changed.

Europe has the second highest dependency of the four regions on imported technology (82%). This compares with 16% for North America, 33% for Japan, and 94% for the relatively insignificant 'rest of the world'.

The concern over the relatively high dependency of Europe on imported technology, compared with its major competitors (North America and Japan), is a major motivating factor for the OMI.

3.1 Microprocessor application areas

The microprocessor market is commonly divided into three sectors:

- Embedded control
- High end workstations
- PCs/low end workstations

The major characteristics of each market sector during 1990-1995 are summarised below. The volumes are for the Western Europe market; this is expected to be about 25% of the World market. The volumes are difficult to predict with confidence. In particular, the increased use of multiprocessing in all areas suggests that the volumes could be much higher.

3.1.1 Embedded control.

(Includes control and monitoring, telecommunications, peripheral controllers, military, consumer, ...) A volume of 1-10M units per year is expected, at a typical CPU price of \$15-30. 16-bit CPUs will continue in widespread use, but there will be an increasing need for 32-bit addressing and 32/64 bit data handling.

3.1.2 PCs/low-end workstations.

A volume of 1-10M units per year is expected, with a price/performance heading towards \$1/MIP; very fierce competition is expected in this market. PC's are moving up and UNIX workstations moving down into this market; all of these products have short life cycles.

3.1.3 High-end workstations.

A volume of 100-500K units per year is expected, at a typical CPU price of \$100. Architectural choices are made for ten year periods; 32-bit RISC choices have been made or are being made now. 64-bit address capability will be required for the early 1990's.

An emerging market is scalable parallel computers. These are seen as a cost effective alternative to conventional supercomputers. High-performance computing plays an increasingly important role in engineering and science, and scalable parallel computers are already providing Europe with an indigenous capability. A volume of 10,000 units per year each having 10 or more processors is estimated world-wide by 1993. The uncertainties in this estimate are large. The acceptance of systems on such a scale will require the establishment of hardware and software standards and the development of advanced programming tools.

3.2 Competitive Situation

Key developments in the marketplace are currently in the 32-bit architecture area, though it is anticipated that this will move to 64-bit in the early 1990's. The dominant architectures in the 32-bit marketplace are the well-established CISC (Complex instruction set) families from the US companies Intel and Motorola.

It is noticeable however that the European-origin microprocessors are well-favoured in Europe as shown in figure 4:

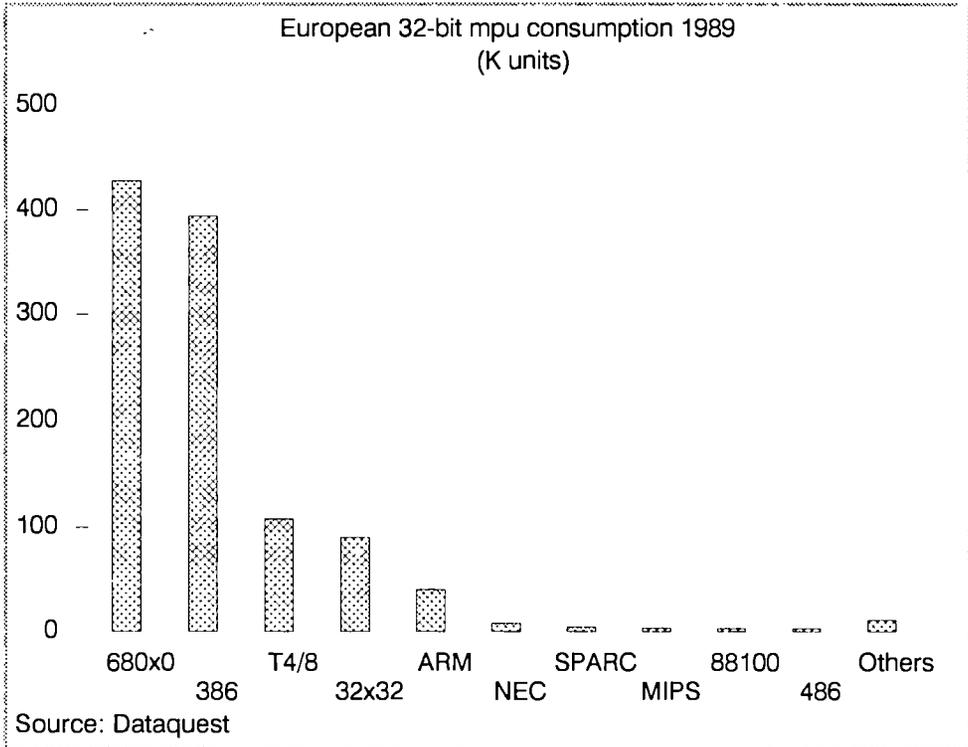


Figure 4 European 32 bit MPU Consumption 1989

Much attention has focussed on the RISC (Reduced instruction set computer) families. These architectures use a simplified form of instruction set in order to gain faster instruction execution, and hence overall improved performance. European-origin microprocessors were the first to be established in this area, and are particularly successful, as shown in figure 5:

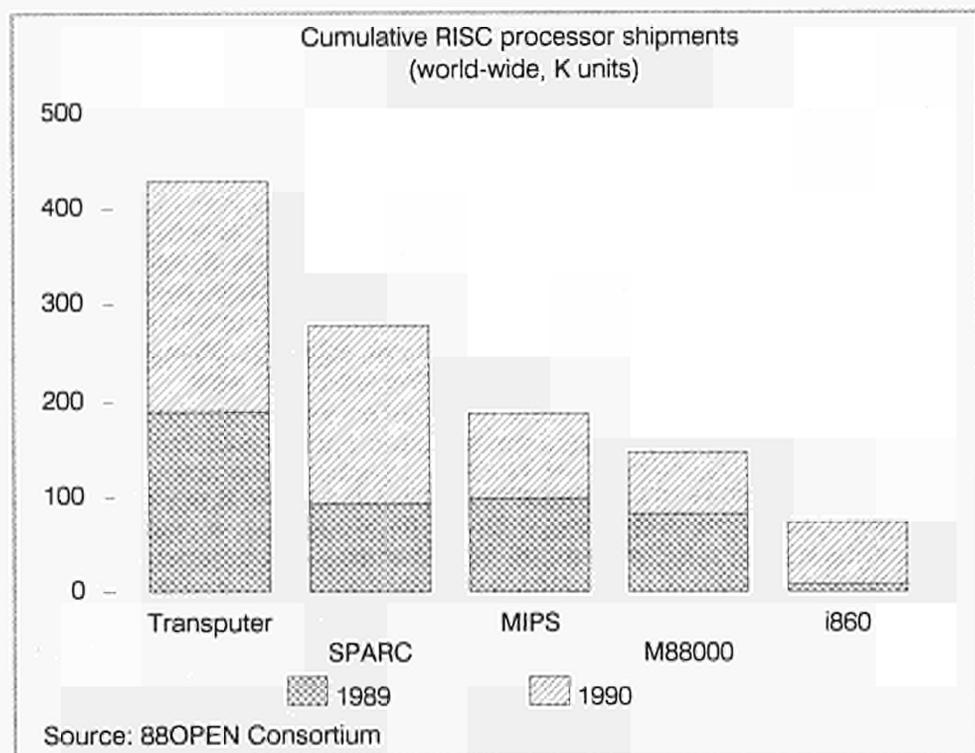


Figure 5 Cumulative RISC Processor Shipments Worldwide

European-origin microprocessors are employed in two of the market areas: embedded control (the Inmos transputer) and low cost workstations (the ARM). The Inmos transputer has unique capabilities for embedded systems, and SGS-THOMSON will continue to further the position of the transputer in this important international market. The ARM also has unique capabilities in that it addresses the low cost workstation market much more effectively than any of the competitive products. It has, so far, been employed primarily in workstations which have gained wide acceptance in the UK education market, and are recognised in a number of international markets. It is marketed by VLSI Technology Inc. worldwide, particularly for application in ASIC macrocell form.

The high-end workstation market is not addressed by either the transputer or the ARM (although the parallel processing capabilities of the transputer have resulted in its use in high-performance computers and accelerators). The demand for standards and compatibility in this market have proved significant barriers to new architectures, although the emergence of the UNIX/C environment has resulted in some standardisation. A variety of the recently introduced RISC architectures (eg: SPARC, MIPS) are currently competing for this market; some of these have been licensed by European semiconductor companies. At the same time, new microprocessor architectures based on VLIW (Very Long Instruction Word) or dynamic instruction scheduling are under development. **It is not sensible to attempt to develop another conventional RISC processor architecture for the workstation market.** The next major opportunity will come after RISC. It is anticipated that new workstation architectures based on multi-

processing will start to appear in about 1992-1994. The need to move to multiprocessing architectures will create an ideal opportunity to enter the workstation market.

4. Requirements for Industrial Applications

4.1 Introduction

This section summarises the requirements of the microprocessor market. This information was contributed by representatives of the European Systems Industry during the European Microprocessor Initiative Applications workshops held in Brussels during the first quarter of 1990. The figures given refer to the European marketplace. The workshops covered Aerospace, Telecomms, Workstations, Office Automation, Large parallel computers, Consumer and Automotive application areas.

4.2 Aerospace

This area has a wide variety of applications within it, but one of its major features is a long development time and a long in-service life. As a result, a stable software environment is important, and lifetime cost, rather than initial cost is the parameter. The ability to develop heterogeneous systems is important in a variety of applications.

High visibility products, as in aircraft, are low volume, high value, but a production run of 800 aircraft may use 200,000 micros, with spares, at 1000 dollars each, ie \$200M over 15 years. The area includes safety critical applications, requiring formal proofs where possible, and fault tolerance. Uses such as satellites have hostile environments and cannot be serviced or repaired. No overall volume numbers are available. No microprocessor component would be designed specifically for aerospace, due to the low volume, but special testing or packaging are feasible due to the high value.

4.3 Telecoms

This is a very important market because it is currently in transition from analogue to digital working, and there is an upsurge in the demand for communications.

The biggest user of micros is seen to be mobile/portable telephones, expected to use 5M units a year, at costs around \$5 dollars (at 1 million off). Networks and PABXs together will also be 5M/year in 1995, 10M/year in 2000, with higher costs, \$10-20. ISDN simple telephones are expected to use 1-2Munits /year. ISDN User terminals are expected to use 500k/year in 1995 to 5M in 2000.

Cost is a driving force in user equipment in the telecomms market, and especially in portable/mobile equipment, the need to achieve high levels of integration is very important. It is expected that these micros will not be off-the-shelf, but would be ASIC designs incorporating processor macrocells.

The need for integrated communications elements is paramount (such as frequency synthesizers, ADC/DAC, timers, keyboard and display controllers). The mobile/portable equipment, as well as line powered user terminals, will need very low power consumption, battery backup, and tolerance of unstable supplies.

4.4 Office Systems

Office systems overlaps with Telecoms, as both claim the cordless phone and both claim the FAX machine.

The Digital Electronic Cordless telephone (DECT) is expected to need 20 MIPS of CPU, 16Kbytes RAM, 128Kbytes ROM, 4Mbit i/o channels to the radio, parallel I/O for keyboard and screen, low power etc, and to be representative of many other portable applications. It will use 100k units / year rising to several millions, and is cost sensitive, needing a very low price (around \$10-20).

The biggest market, other than low end workstations (PCs) will be for printers. Low end printers are very cost sensitive and will remain with their current processors. The major new market is for laser printers interpreting page description languages such as PostScript. Other image handling applications will become important, but laser printers will lead with volumes of several million units a year, with several micros per printer.

4.5 Large Scale Parallel Computers

It is generally agreed that the transputer gives Europe a lead in the exploitation of parallelism, particularly in large scale systems using tens or hundreds of processors. It is expected that work resulting from the PUMA and GPMIMD projects will help to maintain this lead. The market for transputer systems in the 'number crunching' area is currently around 1000 units with 10 or more processors per annum world wide and this market is growing yearly by over 50%. With the availability of the new generation of transputers having a greatly improved performance, it is estimated that this market will grow more rapidly and will increase to 10,000 units per annum by 1993. With this increase will come a growing pressure for standards in support environments, libraries and tools. Such standards are a prerequisite for the widespread acceptance of transputer-based machines and for the continued growth of their market.

4.6 Workstations

The expected volume is 1 M units/year (world wide) in 1995, of which the bulk will be 80x86 compatible, but some 8-10 percent will be high-performance and thus use parallel processing (4-8 CPUs per system). A third of these will be in Europe (ie around 30k high performance systems/year in Europe, 200k micros).

5. Overall Requirements

5.1 Software

For embedded systems, real time capability is important. The major languages used are C and assembler, but other languages such as occam and Ada are needed. Several real-time kernels are in use. There will be an increasing number of safety critical applications needing verified hardware and software.

For PCs/low-end workstations, a large number of off-the-shelf application packages is essential for fast market acceptance; This will be greatly helped by the development of standards such as the Virtual Binary Interface. Many languages are needed (C, Fortran, Pascal, Cobol, etc). Standard operating systems are essential, especially standard UNIX. The processor architectures must support a 32-bit virtual address space per process.

For high end workstations, brand name application packages are required for market acceptance. By spreading their development costs, the Virtual Binary Interface will encourage ISVs to provide these packages early in the Initiative. Efficient support of standard UNIX is required, with extensions for multiprocessing and fault tolerance built

into the architecture. These workstations use embedded subsystems especially for input/output.

5.2 Performance

The application areas divide into three classes:

- those that are price limited and use the performance they can afford, like consumer and automotive;
- those that make a price performance tradeoff, such as office systems;
- those that will use whatever performance they can get, such as military, DSP, image processing, large computers.

Telecoms falls in category one for user equipment, and category two for network/exchange equipment.

Performance also seems to divide into three classes:

- applications that need the level of performance presently available with current micros (often even 8 bit ones), such as TVs and low level printers, and engine control
- applications around 20MIP performance at higher integration; for example all the portable applications of phones and computers;
- applications requiring ultimate performance

Class 3 also divided into two areas, those who want a single processor, and those who are willing to go multiprocessor.

5.3 Integration

This was probably the most emphasised point by all the speakers over the whole workshop series. They require a level of integration that can only be achieved by either collaboration with the silicon supplier, or by an ASIC/Macrocell approach. The high volume users, such as automotive and TV have this collaboration today, with the silicon supplier putting the users choice of standard blocks together for a new version of the 68XX or 80XX families of controllers, or compatibles like the ST6/9 from SGS-THOMSON. Lower volume users do not have access to the necessary levels of integration today.

In the future, both types want the ability to include all the standard I/O cells and peripheral functions, like timers, as well as EPROM, EEROM, and RAM and also a small amount of custom logic.

The custom logic has two functions: it speeds up some important task, and it also makes the chip specific to the one equipment manufacturer, allowing him to differentiate his products and prevent copying.

Market penetration is expected to be eased through the availability of good design tools. Design tools for such ASIC and macrocell facilities are an important topic.

5.4 Cost

Cost savings made when higher CPU performance allows tasks to migrate from hardware to software are often overlooked. Costings were always done on a per CPU basis, not the system cost, although clearly the requests for more integration are in the right direction.

There are again distinct bands acceptable for CPU costs. One does not have to justify them, they are in-built in the minds of the end equipment manufacturer. Consumer applications, including automotive, require \$10 max, \$5 average, or they will not buy. Workstation type applications and larger computers will stand \$100. Aerospace applications will stand \$1000 for flight based systems, but elsewhere they are now aiming to reduce costs by using commercial parts in a protected chassis. Embedded control systems clearly take a spread of performances at a spread of price, but the \$100 price is effectively a ceiling even for compute intensive applications. Note that the designers will pay \$100 for a CPU plus \$200 for its FPU, but for a CPU/FPU single chip, the price is still \$100 dollars, especially in the low end workstation market (PCs). There is a major education task to be done to show buyers they are getting value for money.

5.5 Packaging

Packaging requirements follow on from the desire for greater integration, and in not wasting space saved in silicon in interconnect. Thermal dissipation problems associated with smaller and faster equipment are a key concern.

5.6 Marketing the Open Micro

The problem of software portability must be overcome if the new architecture is to attract a sizeable software base quickly. The adoption of a standard UNIX within the Initiative is a sine qua non. The Virtual Binary Interface offers the prospect of easy migration of software developed outside the Initiative, and of porting software developed within the Initiative between all its architectures. Software written in assembler can only be supported by the use of emulation, and therefore a high performance emulation mode (even if all in software) will be essential also. Note that certain suppliers (INTEL, Motorola...) have a vested interest in preventing such portability in order to protect their captive markets.

Most important of all is the need to ensure that the applications are provided as soon as the silicon so that the new micro can reach volume, and hence low price immediately. To achieve this there must be major funding for application development, be it from CEC funds or from consortia of applications companies. Without this the applications companies each feel they must take the lowest risk/cost route (ie continue to use the architecture/instruction set/training and experience that they already have).

It is accepted by the consumer and automotive industries that they will not be able to use a micro immediately because of price pressure. The silicon vendors need to find a solution to this problem, because these are the industries that can use the volumes necessary to pull the price low for everyone.

6. VLSI technology background

Electronic systems are constructed as a hierarchy of modules. Integrated circuit chips are mounted on modules, with each module containing one or more chips. The modules are then mounted on boards, and a collection of boards is held in a box - which also contains power supplies and cooling. Switching devices - transistors - occur only in the chips.

One of the most obvious features of this technology is that interconnections consume nearly all of the volume - modules and boards contain only wires. Switching devices - transistors - occur only on silicon and themselves consume a tiny fraction of the

available silicon area. Even a semiconductor manufacturer is effectively selling wire, not transistors!

Some significant improvements will come from semiconductor manufacturing processes with more interconnect levels - and from modules and boards allowing more interconnect levels. However, it is likely that wire density - the number of wires crossing a unit area - will continue to be the limiting factor in system construction. It is not the case, as is often supposed, that the number of connection points on a chip is the limiting factor. The number of connection points on a chip can easily be increased to the point where the density of wires needed to route connections away from the chip exceeds the capabilities of the module technology.

Scaling down the size of the devices on a chip has continued for many years and is expected to continue for many more. Present technology achieves transistor gate lengths of about 1 micron. As the device scales down, it is possible to reduce the size of the chips. Provided that the chip is not so large as to have low manufacturing yield, it is much more attractive to use the reduction in device size to put more devices on a chip. In fact, there has also been a steady increase in chip size over the years as manufacturing processes have improved.

Reducing the size of devices on a chip has several effects. A reduction in the size of transistors gives rise to a linear increase in switching speed. However, the number of devices increases quadratically. These two factors provide a good justification for the use of parallelism: the speed of a sequential machine can be scaled linearly but the number of components in a parallel machine can be scaled quadratically.

Although reducing the size of transistors has the effect of increasing their speed, the opposite is the case with interconnections. Signal propagation through a metal track (a 'wire') on silicon is limited in speed by its resistivity and by capacitance effects. Reducing the width of the track increases resistivity and reducing the distance between tracks increases capacitance. The result is that signal propagation delays are increasing relative to switching speeds.

The cost in silicon area in communication between chips is high - about 20% of the area of a transputer is occupied by connection points onto which wires are connected, and there are further costs in area and power consumed by devices to transmit and receive data. Communication between chips over significant distances can be fast - a good transmission line can achieve near light speed ($3 * 10^8$ cm/sec). On chip, the resistance and capacitance effects limit speeds to much less than light speed - about $5 * 10^6$ cm/sec in current VLSI technology.

In the technology used to manufacture transputers, a transistor can switch in about the same time as a signal will travel along 1mm of a metal track. In designing a processor, it is therefore essential that the registers, arithmetic units and control logic are physically close together to ensure a high rate of data transfer, rapid transmission of control signals and rapid distribution of clock signals.

The trends towards lower feature size and larger chip area are expected to continue until at least 2000. Memory technology is used as the driver for developing the manufacturing processes, with complex logic attaining the same levels of complexity at a delay of 1 - 2 years. The following diagram illustrates the main technology parameters expected from 1980 (64K DRAM) to approximately 2000 (1G DRAM).

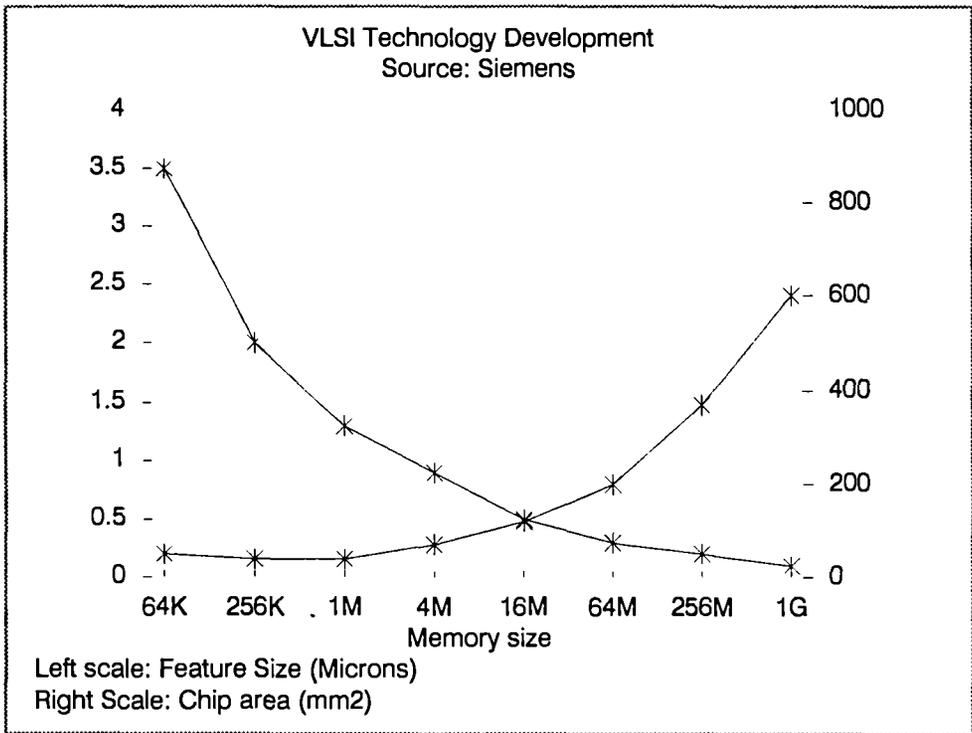


Figure 6 VLSI Technology Development

In order to take advantage of the rapid developments of technology, it is essential that the design of a microprocessor is technology driven. The factors outlined above have the inevitable consequence that the trade-offs made in the design of 80's microprocessor architectures have now to be completely re-evaluated. The highest performance will come from the exploitation of parallelism in the implementation of apparently sequential instruction sets, and the use of multiprocessing, both on-chip and between chips. The OMI architecture has to allow the full exploitation of technology, whilst permitting lower performance low cost variants.

7. Architecture baseline

The starting point for the OMI microprocessor architecture work is a combination of licensed-in non-European technologies and existing European technologies and know-how. It is important to expand and consolidate the existing worldwide position held by Europe in this technology driven market to ensure the success of the new architecture.

7.1 Licensed architectures

Several US architectures (SPARC, MIPS, 88000) have been licensed by European manufacturers in order to use them in standard microprocessor components or as macrocells.

It is intended that the OMI should provide interfaces and tools to support the use of these processors in heterogeneous systems and should support their use as processor

macrocells. This will allow Europe to exploit non-European technology where appropriate and, conversely, will facilitate the export of European microprocessor technology.

7.2 Novel architectures

It is inevitable (and highly desirable) that the development of innovative architectures should continue within Europe. Many of these will be application specific architectures, although an interesting example of a general purpose architecture is the high speed bipolar RISC processor developed by PGC Limited to allow real-time emulation of standard architectures.

To facilitate the rapid introduction and exploitation of new architectures, it is intended that the OMI should provide interfaces and tools to enable them to be used in heterogeneous systems.

7.3 The transputer

The current transputer architecture was designed as a general purpose component for high performance embedded processing. The major characteristics are:

- No operating system required (microcoded kernel easily outperforms one)
- Scalable parallel processing in specialised processor configurations (eg arrays with nearest-neighbour communications)
- Explicitly allocated processing, memory, communications to guarantee real-time performance

7.4 The ARM

The ARM was designed with the objective of providing a very cost-effective general purpose workstation processor suitable for implementation using standard manufacturing processes. It will continue to form the basis of a range of low cost workstations supporting all the standard operating systems (UNIX, MSDOS through emulation, ...).

8. How to meet the needs

The strategy proposed is for the *next* generation of IT products to be addressed by a new architecture based on *general purpose* multiprocessing. This architecture will form the basis for the embedded systems of the late 1990s, and will also be appropriate to the workstation market. The strategy for the new architecture is as follows:

- A new CPU and system architecture will be developed to address the general purpose multiprocessing market; the first products using this architecture will be introduced in 1994-5.
- The components of the new architecture will be made available in macrocell form, using the tools developed to support embedded system development.
- In the major embedded system areas, applications demonstrators will act as a focus for the development of silicon and software components.
- A major programme will be established to develop workstations and multiprocessor computers - and their associated systems software and applications software. This will start as soon as possible after the new generation architecture is specified.

- A major European initiative will address the launch of the new workstation. The best initial target is the higher education market. This market is receptive to new ideas and has been the source of much software development.

8.1 The new generation

The new generation of microprocessors will be the first to support *general purpose* multiprocessing. These products are important for three reasons:

- They provide an opportunity to enter a new market. This is a major opportunity: multiprocessor computers consume more processors than sequential ones. It seems likely that general purpose multiprocessors will dominate the high performance computing market in the late 1990s.
- General purpose parallel computers will start to be employed in embedded systems - they will allow standard multiprocessor systems to be mass produced on big chips.
- They will offer higher performance than the current generation and this will continue to be important in the embedded systems market.
- General purpose parallel computers will be essential to the development and simulation of specialised high performance systems.

Multiprocessing offers a new opportunity in the workstation market - but it is *essential* to minimise the difficulties of product introduction by developing an architecture which:

- Supports existing software as far as possible - this means UNIX (or an accepted standard derivative of it), and standard languages (with parallelising compilers)
- Supports parallel versions of existing languages (and novel ones such as LISP, PROLOG, C++ etc.)
- Supports environments for the development, release and maintenance of software
- Provides scalable parallel processing with processing/memory/communications suitably balanced for *general purpose* parallel processing
- As far as possible hides the complications of dealing with parallelism (using virtual processing, virtual memory, virtual communications)
- Has a 64 bit address space

8.2 Processor cores as macrocells

For many applications, the key challenge of the 1990's is achieving new levels of integration. The technology will permit a high performance processor, a substantial amount of memory, communications, interfaces and application specific logic to be integrated onto a single chip. Success in exploiting this ability is seen as a key requirement to maintaining competitiveness. The new architecture should facilitate the development of processors as macrocells to be integrated into ASIC libraries. Standard processors are required in order to avoid the need for the development of special-purpose system software. It is highly desirable that the same architecture should be exploited both as standard microprocessor components and as processor macrocells.

Although complete processors will be offered as single macrocells, it is also intended to make the component modules of processors available as macrocells. These macrocells can be used as the basis for high level silicon compilers which generate specialised processors from behavioural descriptions (eg in VHDL or occam). This will allow fast, reliable design of *concurrent silicon systems* consisting of networks of standard general purpose processors together with specialised processing elements.

Silicon compilers also offer an effective technique for progressively migrating software functions into optimised silicon implementations.

8.3 Design environment

Design tools appropriate to the development of mixed hardware and software systems must be provided. This area is not adequately addressed by any existing set of tools and is likely to be a major barrier to the use of processor macrocells. It is believed that Europe is well placed to develop appropriate tools in this area. There is a need for European systems companies to provide support to companies making use of such capabilities.

In order to make use of processors and modules as macrocells, an appropriate multifoundry library must be provided. Corresponding proprietary (and standard) CAD interfaces, and software components (including testing, hardware and software debugging) must also be provided. These macrocells will be made available on state-of-the-art foundry technology; this should involve appropriate links to JESSI.

For safety critical applications, verified implementations of processors are needed. These must have rigorously specified instruction sets (which are the basis for verified compilers) and interfaces (which allows the verification of the complete hardware system).

It is proposed that the correct strategy is to commence the work to develop the necessary CAD tools immediately, using existing processors for technology pull-through. Any such tools should be designed to be portable between a range of CAD environments, and should be capable of supporting a range of processors.

Both of the existing European architectures are well suited to macrocell implementations. They are relatively simple designs and it will be possible to construct silicon systems incorporating one or more processors together with other specialised designs. The most obvious rivals all consume a relatively large silicon area as they were designed to optimise performance using state-of-the-art technology.

The effect of initiating this activity immediately is that effective macrocell technology will be available throughout Europe allowing rapid transfer of the new generation to the macrocell library.

8.4 Software support for the new generation

The new generation processors will be supported by software development environments compatible with existing standards (PCTE, SQL, C, POSIX ...) and development methods. The OMI will draw on other projects to provide an environment which integrates the various tools and provides software development monitoring, delivery and maintenance. A key feature of the OMI is the support of libraries and the transfer of software modules between application projects.

8.5 Software portability from existing architectures

The new generation architecture will be developed with the goal of minimising software migration costs from previous platforms. The ability to migrate software is essential for the PC and workstation application areas, and is increasingly important in some embedded system areas. Many embedded systems are developed for specific applications, but a significant (and growing) number make use of software initially

developed for workstations and PCs. An obvious example is the use of embedded postscript processors in laser printers.

The porting issue will be eased by the widespread acceptance of standard versions of UNIX, which will also be supported by the new architecture. However other machine dependent issues affect the ease of porting, such as the machine environment (eg memory and disc sizes, organization of video frame buffers) and the basic facilities (eg support for half-word types, number of registers, virtual address space). The new architecture will be designed to avoid compromising any of these issues.

In order that the systems houses' existing investments in applications are preserved as far as is possible, it is proposed that the architectural definition phase of the project include activities within the research and development functions of major European systems integrators to define the factors which affect porting costs most strongly. This would provide valuable input to the architects to improve the eventual market acceptability of their design.

Various mechanisms have been proposed to ease the software porting problem:

- A virtual binary interface (VBI) allows the distribution of software in a form that could be run on any platform equipped with an installer. A similar proposal has been made by the OSF, called the Architecture Neutral Distribution Format (ANDF) and a European technology chosen. The widespread acceptance of such a standard would greatly simplify the introduction of a new architecture.
- Source recompilation. This is the normal method of application porting at present, and is usually complicated by compiler and machine environment differences. Good support for configuration management is required in order to perform it correctly.
- Execution of existing binaries by software emulation of old architectures. This is used at present mainly to allow DOS applications to run in a UNIX environment. Special hardware can make the emulation faster at some additional cost. This method usually consumes considerable machine resource (memory and MIPS).
- Recompilation of the existing binary. This is less efficient than recompiling the original source.

Each of the above methods has advantages and disadvantages in terms of development costs, porting costs, and requirements on machine resources and distribution media. The optimum solution(s) must be established in collaboration with the customers.

8.6 Migration path from existing architectures

A key objective is to provide a migration path to allow designs based on existing architectures to be upgraded to take advantage of the new generation. This will be achieved by developing interfaces between the various architectures.

8.7 Interfacing the different architectures

The potential benefits of developing standard interfaces between the various architectures are considerable. An increasing number of application areas require the ability to mix general purpose processing (UNIX etc) with high performance parallel processing. An obvious example is office equipment. This can be achieved by a combination of, for example, an ARM (to support UNIX) and a transputer network (to provide high performance embedded processing).

8.8 Hardware interfacing

The major requirement in this area is the ability to connect the Inmos link system to the various different architectures. The transputer already has the capability to provide multiprocessor systems; by designing a transputer link interface for other architectures it will be possible to mix architectures in a much more closely integrated manner than is currently practised. This requirement has already been identified in existing office-equipment projects which combine (eg) the ARM and the transputer, and in existing products which combine (eg) the SPARC and the transputer.

To allow connection to standard workstations, interfaces to standard busses such as VME, Futurebus+ and Microchannel must be provided. Ideally, these interfaces should be fast: it is desirable to give attached processors access to the graphics system and disk system.

For macrocell applications, it is not necessary to interface to standard busses, but it is desirable to construct fast interfaces onto (eg) SPARC and MIPS core processors.

It is proposed to develop fast interfaces between the Inmos link system and these other architectures. This will allow interworking between the architectures at the hardware interface level.

Note that transputers are already in use to provide message routing; further components are under development to support high-speed low-latency message routing in hardware. *These capabilities are as important to the construction of multi-processing systems as the microprocessors themselves: it is possible to create standards based on European technology in this area.*

8.9 Software interfacing

It is essential to provide effective software interfaces between the various processors. Important requirements are:

- A standard binary data format (binary instruction compatibility is of course not possible - and is not necessary)
- The ability to use standard programming languages with standard applications support interfaces to program heterogeneous systems.
- The ability to interface embedded parallel processing systems to processors providing standard operating system services (eg UNIX).

A preferred standard serial binary data format will be defined for the link system. This will cover formats for basic data types and structures.

An open consortium will be established to facilitate widespread adoption of the link system.

The expected benefits of enabling the various architectures to interwork are:

- Workstations and PCs based on non-European architectures are widespread. They are used for program development - both for embedded systems and for high performance parallel computers. Often, the value of the attached (specialised) parallel processing equipment exceeds the value of the workstation.
- It will facilitate the smooth migration of system functions between the different architectures.
- It will allow the construction of heterogeneous systems incorporating both specialised processors and general purpose processors.
- The market for embedded subsystem components in future general purpose computers is substantial.

8.10 Software support

It is to be noted that the American leadership in the domain of software packages can be explained by an easier and earlier access to the internals of the machines under development. This leading edge can be similarly gained on the OM machines by providing a proper development environment to software product developers.

Continuity with the conventional software development world is needed to favour software migration as well as to support established development methods.

8.11 Applications development

A key feature of the OMI is the 'vertical integration' of applications with software development, CAD tool development and silicon component development. Much of the work will take place within applications projects, which will produce demonstrators and form the environment for the development of relevant infrastructure. This is particularly important for the development of processor macrocell techniques, and hence of high performance application specific integrated circuits based on standard processors.

9. Establishing the European Workstation

Workstations and computers constitute a major strategic market. However, to establish the new architecture in this market it is essential that a user base for platforms and workstations is developed rapidly. For the low-end workstations, the education market is a natural market - and one familiar to Olivetti/Acorn. Similarly, for the high-end workstations, the academic/research market is an ideal starting point. The OMI workstations should be installed in all European tertiary education institutes.

10. Standards and IPR

Standards will be promulgated publicly (eg: Virtual Binary Interface, Data format, Communications protocols). Product IPR will rest with the developing companies. Architecture, instructions sets and interfaces will be protected; licences will be available subject to normal commercial negotiations.

11. Conclusions

An Open micro can be developed to take a major share in all markets. However, to address the Low end Workstation/PC area, a mix of efficient emulation and the use of VBI is required so that off the shelf shrink-wrapped applications can be run. Consumer and automotive applications will not develop unless the price is set very aggressively from the outset.

To enable such a micro to sell in sufficient numbers from day one to support the low cost needed and still allow the silicon vendors to operate profitably requires that there be applications with high volume needs ready to go from day one.

To achieve this requires real end-user applications, not just (very necessary) software tools, to be developed in parallel with the silicon. These must be from volume vendors, and not mere demonstrators. The risk for ISVs will be substantial. A mixture of measures such as the formation of large consortia, risk reduction (e.g. through the use of the VBI) and OMI funding will certainly be required.

Microprocessor Architecture Project.

12. Background.

This project was set up under Esprit 2 funding to survey selected application domains and determine their technical requirements, to investigate and develop major techniques needed for the launch of a coherent architecture, to ensure that the architecture can be implemented by more than one source, and to provide an architecture proposal.

During the development of an architecture, there comes a point where there has to be a commitment; at this point, if further problems are found, it is often not possible to go back far enough to overcome the problem, without major implications on the timing. If a high-tech product is 6 months late to market but on budget, the product will earn 33% less over 5 years, whereas if it is launched on time, but 50% over budget, there will be only a reduction in profit of 4%, according to a scenario developed by McKinsey & Company, an international management consulting firm [1].

Some conclusions can be drawn from this:

- The architecture must be right first time.
- The architecture must be flexible.
- A commitment has to be made relatively early, as prolonged studies will cause it to miss the opportunity.

This is an initial path-finding project, and is expected to indicate the direction in which future developments should be orientated. In particular, early identification of market requirements and an outline as to how these can be met will be of enormous benefit in compressing the timescales that are needed today.

Some of the early results and part of the technical strategy are given here.

13. Problems to be solved.

What are the key problems involved in bringing a new architecture to the marketplace and how can they be overcome?

To be successful, not only does the hardware have to be in place at the right time, but support tools are needed such as compilers, debuggers, and an operating system, to name a few. In addition, applications need to be there immediately the hardware is available. This implies that porting from one system to another is trivial, and claims made for this in the past have not always been borne out e.g. 680x0 based Sun workstations to Sparc based Sun workstations. Another possibility is for the hardware providers to work closely with the application manufacturers; again, there are not many precedents - PCs using the latest 80x86 from a few favoured manufacturers are the main example. These represent the twin problems:

- compatibility
- co-operation

14. Compatibility Issues

This can be subdivided into two cases - where there is no access to source code and where there is access to source code.

14.1 No source code access.

This is often the case for mass market products such as software for PCs. The unavailability of source code is often to protect IPR (Intellectual Property Rights), as the size of the market makes piracy extremely attractive financially. Thus it is necessary to approach the originator of a piece of software and negotiate for a port. This leads to another problem - suppliers of software to the mass PC market are only interested in porting if the user can demonstrate a large installed base, often around the 500 000 mark. Thus, to introduce a new architecture, the vendor has to convince 500 000 users to buy it without most software so that the software houses can be persuaded that the market exists for a port of their products. Put another way, it is like saying that you will run commercial flights to the moon only when there are half a million people there waiting to come home!

There is a way to break through this problem, and that is with emulation techniques. Software emulation has often concentrated on the full emulation of a processor and its' entire instruction set; the result of this is that it runs slowly, at a tenth of the speed of the host processor if it is a good implementation. An alternative technique, pioneered by Acorn and extended within this project is to provide a small piece of additional hardware alongside the processor to emulate some of the key features of the processor to be emulated, and to provide the rest of the emulation in software. Furthermore, if the emulation is concentrated on emulation of the target processor in a particular environment e.g. in the PC, it can be made more efficient by translating a stream of instructions to a different stream, thus emulating the function rather than the detail. In a PC, there are a lot of basic functions that are called via the BIOS (Basic Input Output System), with the video subsystem and network subsystem extending these. So, development of emulation techniques can be used to provide acceptable performance for existing software where source code is unavailable, but a lot of effort is needed, so it is mostly an applicable technique to the mass market.

14.2 Source code access.

Access to source code in a more limited form is usually possible for systems that are shipped in lower volume than PCs and in many more diverse forms. Workstations fall into this category. There is still the problem for the software originator of protecting the IPR. A recent development, ANDF (Architecture Neutral Distribution Format), also known in Europe as VBI (Virtual Binary Interface), is a method for breaking the compilation process down into two parts, namely the *producer* and the *installer*. The *producer* is used by the software originator to compile the code, but the program is compiled down to an intermediate code which retains the information necessary for optimising for different target processors. The *installer* is "owned" by the end user, and this is used to convert the intermediate code into the binary that runs on the machine. This has a number of advantages, in that for m languages and n target processors, only m producers and n installers are needed, in place of $m * n$ compilers in the traditional way. In addition, this removes the need for source code access and still allows distribution of programs where the target is unknown.

To bring a new architecture to the marketplace, all that is required is an installer, and immediately a range of languages and applications are available. Europe is a leader in this technology, and intends to exploit this in the project.

15. Co-operation.

Esprit provides a well known, tried and tested framework for co-operation in working together at the precompetitive stage. This will need to be extended for the future if the goal of multiple sourcing of the architecture is to become a reality. There is a trend in the semiconductor industry towards total cross-licensing deals, lasting for a number of years. SGS-THOMSON and Siemens are two companies in Europe who have agreements of this sort. --

16. Technology.

It is expected that the technology will come from the JESSI projects, and that the OMI is an implementation of this technology. The requirements of processor design and memory design differ. The die size of a processor chip is more dependant than a memory on the interconnect technology; memories require "one to many" (or bussed) interconnections, whereas processors use a much higher percentage of "point to point" interconnections. Use of technologies such as three layer interconnect is vital in keeping the die size down, and hence reducing power consumption and increasing speed.

Power dissipation has become a major factor, and some of the latest processors are becoming limited by this more than anything else. With a trend to portable computing, with its very strict requirements for low power consumption, techniques are being investigated for reducing the power dissipation. There is an industry trend towards 3.3v as a means to lower power consumption, but there are other techniques. Asynchronous logic, where the logic is not driven synchronously by a global clock, offers useful techniques. Reduction of ground and Vcc noise on-chip will occur if not every gate is switching at the same time; this will allow thresholds to be further reduced. For the use of asynchronous logic to become widespread, there are some important user perceptions to be overcome; for example, are users ready to accept that the speed is in a given range rather than being precisely determined and repeatable? With temperature being a governing factor in the speed, will users be happy if their system with its 100 000 Dhrystones gives 99 500 Dhrystones on a warm day and 101 000 on a cold day? Or would the user prefer to have 90 000 Dhrystones every day, hot or cold? If asynchronous logic is going to be a necessity in the future, then we must start preparing the market now to accept situations like this.

17. MAP Architecture.

17.1 Overview

The MAP architecture is being designed to exploit technology from 1994 (10^7 - 10^8 transistors); current processors are $2 * 10^6$ at the leading edge of technology development. Driven by time to market, the major factor in market penetration and profitability, the architecture has to be modular to ensure fast design at this level of integration.

When referring to high performance computing today, it is now assumed that there is some degree of parallelism in the system, and so efficient support for this in a way as transparent to the user as possible is necessary. Today, if your spreadsheet runs too slowly, you go out and purchase a floating point co-processor; this represents a step change in performance for applications that make intensive usage of floating point, but would not speed up a database application. The next logical step will be that when the application does not run fast enough, you purchase an additional processor card

and plug it in in the same manner, but when the performance is inadequate again, further processor cards can be installed. Thus, it is essential that the architecture is a scalable multiprocessing architecture and combines the best features of common memory and message passing architectures, the two main programming models.

The architecture is being defined in terms of:

- communication protocols (Serial Binary Interface Standard)
- memory access protocols
- virtual binary (to support software distribution and heterogenous systems)



This allows a variety of implementations:

- Fast sequential processors using multiple function units for high performance applications
- Simple low power processors for portable applications
- Dedicated Real-time processors and programmable hardware for i/o
- Specialised co-processors, such as those for vector and graphics processing

17.2 Communication Protocols

The Serial Binary Interface Standard [2] will support communication in heterogenous systems. Several different uses are possible:

- Communication between machines with different architectures, but executing processes written in the same language (where two machines may represent the same data type in different ways e.g. the problem of operating on bytes in a word in C, where some expect a little-ended implementation and others expect a big-ended implementation).
- Communication between processes written in different languages.
- Communication with hardware implemented modules.
- Passing parameters between the modules of a distributed kernel or operating system.

Data objects will be converted into the standard representation on output, and converted from this representation on input. The standard does not constrain the representation of the data in memory.

The standard under development is being designed to the following principles:

- The serial representation is strictly little-endian.
- Each data object is represented using $\delta \times n$ bits.
- Data structures are packed to optimise the use of communication bandwidth. Packing values of length $\delta \times n$ is optimal; packing of other values in some cases will not be optimal, in order to minimise difficulties in translation.
- The two communicating processes will agree on the type of data being transferred.
- On-the-fly translation between internal and external representations are possible.
- Each data object is transferred as a single message. Where translation is performed by software, it will often be necessary to use an intermediate buffer, which may be implementation restricted.

At present, two levels are anticipated - level 0 and level 1.

The level 0 standard supports the following data types:

	C	FORTRAN77	occam
boolean	bool	LOGICAL	BOOL
byte	char	CHAR	BYTE
16-bit signed integer	short	INTEGER*2	INT16
32-bit signed integer	long	INTEGER	INT32
32-bit IEEE floating	float	REAL	REAL32
64-bit floating	double	DOUBLE	REAL64

The level 0 standard also supports one dimensional arrays of any of the above types. The level 1 standard adds multidimensional arrays and records, and applies mostly to C and occam.

One way to implement the binary standard for C and FORTRAN programs is to provide procedures which input and output data types according to the standard. These procedures would all have two parameters; one defines the data object to be communicated and the other is the object, or a pointer to it.

17.3 Processor Architecture

There is a need to maintain a state of the art design to be competitive and provide flexibility in the different implementations to cover a wide area of applications. To do this, a modular processor architecture is envisaged.

The processor executes instructions from a collection of istreams (instruction streams) [3]. The instructions in an istream are executed sequentially. Each istream is independent of the others. An istream function unit deals with fetching and decoding of instructions, fetching of operands, execution of all jumps, conditional and unconditional, calls and returns and writing back the results obtained from other function units. Other operations such as integer and floating point arithmetic are performed by other function units.

In a low cost implementation, there would be a single istream unit; a higher performance (and cost) implementation would have multiple istream units, able to keep a number of ALUs and FPUs busy concurrently. This allows latency in one istream to be hidden by executing instructions from other istreams e.g. in the event of a cache line miss.

A scheduler unit is required to deal with scheduling, descheduling, running and synchronising of istreams.

17.4 Instruction Set

One of the disadvantages of the traditional RISC instruction set is the lack of compact representation of code; in turn, this places an undue burden on the memory bandwidth, resulting in separate data and instruction busses for many implementations requiring highest performance. This makes low cost implementations difficult, as more memory chips are needed.

CISC processors and the existing range of transputers use a variable length instruction set; the difference between the two is that traditional CISC processors have a minimum instruction length of a *word*, and the transputers have a minimum instruction length of a *byte*, thus giving up to four instructions in a 32 bit word.

The instruction set under development uses variable length instructions to minimise the memory bandwidth required by having a compact instruction representation.

17.5 Memory and Communications Architecture

Communication of data, whether to other processors or to peripherals, is becoming ever more important, to the extent that some computer manufacturers are considering creating standards in this area. What is also clear is that a universal parallel programming paradigm is required for programming single processors for multitasking and multiple processors in parallel computers and heterogenous systems. There are two basic models in use today - common memory (also known as shared memory) and message passing systems. The former work well with small numbers of processors, but do not scale; the latter probably require more programming effort, at least to the uninitiated, but are the only efficient method for tens of processors upwards.

It is possible to use a common memory model, with the implementation riding on top of the message passing architecture. This is known as Distributed Common Memory, or Virtually Shared Memory. Work is underway in the project to define and simulate this, prior to the implementation of all or part of the required primitives in hardware. This follows Valiant's BSP (Bulk Synchronous Parallelism) model[4].

Work is also underway to consider a scheme where the communication mechanism is independent of the hardware, allowing wide communications e.g. byte or word wide, at high speed and throughput, or a serial communications mechanism with lower real estate i.e. pin count between chips (inter-chip) and silicon area on chip (intra-chip). This will allow the communications bandwidth and cost to be traded in any given system implementation. The serial binary interface standard can be extended to become a communications binary interface standard.

18. Summary

The outlook for the success of bringing a new architecture to the market is high; many pieces of the technical jigsaw, which have been missing in the past, are now in place. Most of this baseline has already been developed in previous co-operative projects, so that is not an imposed initiative, but one with a vision by those concerned to build upon the solid achievements to date to achieve the necessary critical mass for success, in which will all share and benefit from.

19. References.

- [1] Megatrends 2000 p254.
- [2] Serial Binary Interface Standard - David May; confidential project document April 5, 1991.
- [3] E1 Instruction Execution Architecture - Andy Sturges, Nathan Sidwell; confidential project document April 5, 1991.
- [4] A Bridging Model for Parallel Computation. Leslie G.Valiant; Communications of the ACM, August 1990. Vol.33,No.8.

Nanoelectronics: A Vision of the Future?

Steven Beaumont
Nanoelectronics Research Centre
Department of Electronics and Electrical Engineering
University of Glasgow
Glasgow G12 8QQ
Scotland, UK

SUMMARY

This article gives an overview of the study of Nanoelectronics and describes some of the aims and results of three Basic Research Actions in this area.

1. Introduction

As undergraduates we are taught that an electron behaves as a wave as well as a particle. Yet as we continue to learn about electronic devices we rapidly lose sight of this wave character and think of the electron as a particle. This is because the dimensions of a transistor are just too big for typical phenomena associated with waves, such as diffraction and interference, to be important. We only observe the diffraction of light, for example, when it passes through slits whose widths are comparable to the wavelength. Since the electron wavelength in a typical semiconductor is on the order of a few tens of nanometres ($1\text{nm} = 10^{-9}\text{m}$), we can only expect to meet electron wave effects when we carry out experiments on this size scale. In addition, the operating conditions of a transistor are such that the electron's associated wave changes wavelength and phase very rapidly as it is scattered by the lattice and by impurities.

As our ability to make smaller and smaller structures and devices has improved, however, we have begun to be able to observe phenomena in semiconductors which are directly attributable to the electron as a wave, rather than as a particle. The study of these effects and the devices in which they arise has become known as 'Nanoelectronics'.

2. Quantum confinement in grown structures

The first nanoelectronic devices to emerge on the market are the High Electron Mobility Transistor (HEMT - otherwise called the Modulation Doped Transistor, or MODFET) and the Quantum Well Laser. HEMTs are found in the front ends of satellite receivers, and Quantum Well lasers are beginning to appear in CD players. Some people may be surprised to find these devices referred to as 'nanoelectronic', in the way in which I have defined the term: but it is quite justifiable. Both are fabricated using very precise epitaxial growth techniques to form stacks of different semiconductor alloys. Whilst the crystal lattice of these alloys match perfectly, the energies of the electrons in each component are different. Consequently, electrons injected or doped into these structures prefer to occupy the alloy with the lowest available energy states.

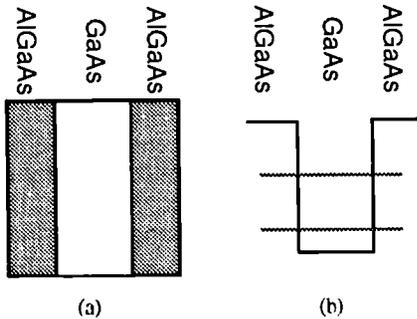


Figure 1. Layer structure (a) and energy (conduction band) diagram (b) of an AlGaAs-GaAs-AlGaAs quantum well. The dotted lines indicate the energies of the confined states.

Figure (1) shows the energy diagram of a layer of gallium arsenide (GaAs) sandwiched between two layers of aluminium gallium arsenide (AlGaAs). If grown properly, the interface between each layer is atomically smooth, switching from one alloy to its neighbour in a single atomic plane. Consequently, we can make the GaAs 'well' very thin - so thin, in fact, that it can confine the wave associated with any electron that tries to occupy it. Most physicists and electronic engineers are taught about this situation when they study elementary quantum mechanics. One finds by solving Schrödinger's equation for the well that electrons cannot reduce their energy to the level of the bottom of the well: the lowest available energy state is somewhat above the well's 'floor', to an extent which depends inversely on its width. As the well gets thinner, so the lowest energy state the electron can occupy moves to higher energies. These energy states are referred to as sub-bands and they arise directly from the confinement of the electron wave in the direction in which the original semiconductor layers were grown. There is a series of these subbands at higher energies in the well, and an electron in one of them can only gain energy by moving along, not across, the well. An electron in the lowest subband is therefore only free to move in the plane of the layers - it has just two degrees of freedom (2D) compared to three degrees of freedom in bulk, homogeneous semiconductors.

How is confinement by growth useful in real devices? In the quantum well laser, the ability to vary the lowest subband energy by changing the thickness of the well means that we can tune the colour of light emitted from the laser. Light emission occurs when an electron recombines with a hole and the wavelength of light emitted depends on the energy difference between the particles. Since we can tune the electron's energy using the thickness of the well, we can tune the light emission. There are other benefits, too. In a 2D system the density of electrons concentrated around the bottom of the first subband is much greater than in bulk material, so more electrons are available to combine to emit the wavelength of light desired. In bulk semiconductors this is not so, and light is emitted over a much broader spectrum.

In the layers in which HEMTs are fabricated, one or both barriers either side of the quantum well are doped, but once electrons are released from their parent impurities, they fall into the well and are confined there. Again, quantum effects determine the energy states they can occupy, and motion is constrained to a two dimensional plane perpendicular to the growth direction. It is this confinement that is important for the HEMTs operation as it keeps the electrons separate from the donor ions so there is less scattering of negatively charged electrons by positively charged donors. The mobility of the electrons is therefore greater than in bulk, uniformly doped semiconductor. In addition, there are large numbers of carriers confined in a thin layer, which produces a device with a large transconductance - the output current is very sensitive to small changes of gate voltage. Finally, the confinement is good even when the electrons are accelerated to high velocity by large electric fields, as is usual in normal operation. This gives the HEMT a low output conductance. Overall, the HEMT has high gain, low noise and, moreover, is very fast. The first two properties arise from the direct

or indirect effects of electron confinement. The third, however, depends on the length of the gate of the transistor.

The cutoff frequency of a HEMT depends inversely on its gate length: the shorter the gate, the higher the cutoff frequency. If the gate can be made very short, the speed should increase correspondingly. Nanoelectronic technology has more tricks up its sleeve than the growth of precision layers. Electron and ion beam lithography techniques allow one to write patterns on the surfaces of HEMT and quantum well layers which can be as small as 10nm in size - almost as small as the dimensions defined by growth. A transistor made with a gate this short would be able to amplify current at frequencies up to 1THz (10^{12} Hz)! Is this possible? One of the ESPRIT nanoelectronics projects, 'NANOFET', is trying to find out by making pseudomorphic¹ HEMTs with the shortest gates allowed by electron beam lithography. Apart from high resolution lithography the project has had to develop methods of recessing gates without loss of resolution, and has also needed to optimise the epitaxial layer structures required for the project. In parallel with their experimental work the consortium has studied models of short gate transistors which suggest that, whilst THz frequencies might not be accessible because of parasitic effects, frequencies of several hundred GHz ($1\text{GHz} = 10^9\text{Hz}$) may very well be achieved. One important question for the wider community interested in such high speed devices is how they are best tested. Conventional network analysers are not up to the task: perhaps assessment by picosecond optical stimulation and detection of voltage pulses is the most appropriate method at the moment. The work of the NANOFET consortium on this matter should provide valuable experience.

3. Quantum confinement in laterally patterned devices

Short gate transistors are far from being the only interesting area of study to emerge from the combination of high resolution lithography and precision growth. I have already remarked on the high electron mobility attainable in HEMT layers at room temperature. At low temperatures, the improvement relative to bulk material is even more remarkable.

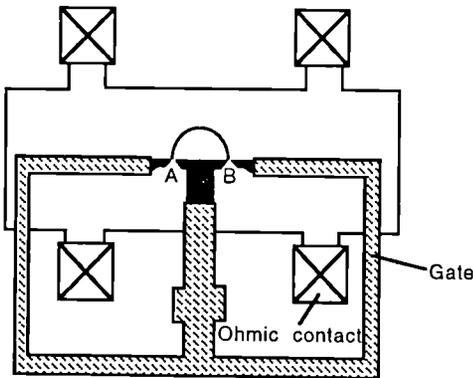


Figure 2. Device used to demonstrate solid state electron beam formation

A properly fabricated HEMT layer can have a mobility of $10^7\text{cm}^2/\text{Vs}$ at 4K and yet still contain $\sim 10^{11}$ electrons per square centimetre: this means that the electrons can travel for many μm ($1\mu\text{m} = 10^{-6}\text{m}$) before scattering, changing its energy and its phase. This *ballistic transport* behaviour is reminiscent of the flow of electrons in a vacuum tube. We could even think of making devices based on the formation of electron beams in the solid state, which could be steered from one circuit to another using gates to deflect the beam. Members of the Esprit Basic Research consortium 'NANSDEV' have managed to demonstrate the principles of such a

¹ A pseudomorphic HEMT is made with a channel of InGaAs sandwiched between GaAs and AlGaAs. Because InGaAs has a different lattice spacing to its neighbours, it is only by virtue of strain that the mismatch can be accommodated. The InGaAs layer must be thin ($\sim 15\text{nm}$) otherwise dislocations will occur which will seriously degrade the properties of the device.

device. Using the electrode pattern shown schematically in Figure (2), a beam was formed by extracting electrons from aperture A and bending it round using a magnetic field into aperture B. Connoisseurs of electron optics will recognise this device as the solid state analogue of the vacuum electron spectrometer and indeed some of the properties of the vacuum device, such as the focussing of the beam by the deflecting field, are exhibited in the solid state version.

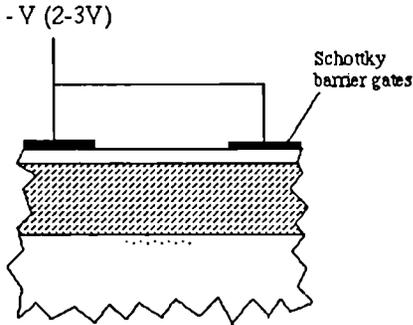


Figure 3. Cross section through a gated point contact device

The injection and collection apertures used in this device have even more interesting properties. Figure (3) is a cross-section through one of these apertures (more commonly referred to as *point contacts*). As the surface gates, which are separated by no more than about 0.25 μm , are biased negatively, electrons underneath them are depleted and eventually a narrow channel is formed in which carriers are confined laterally by the electrostatic potential as well as vertically by the heterojunction barrier. Instead of a 2D system, we have further reduced the electron's degrees of freedom to just one (1D). By increasing the voltage on the gates, we can reduce the width of the channel until eventually it will disappear completely. As the wire becomes confined laterally as well as vertically, the subbands due to the vertical confinement split into lateral subbands, whose separation again increases as the wire width diminishes. Pioneering experiments by members of the NANSDEV consortium, and by members of LATMIC, the third ESPRIT Basic Research project on nanoelectronics, showed that the conductance of a 1D channel formed in this way is quantised in units of $2e^2/h$, where e is the electronic charge and h is Planck's constant. This effect is due to the fact that each lateral subband adds the same conductance to the channel as it becomes occupied with electrons, and is direct evidence that the electron can be squeezed laterally into a channel of comparable width to its wavelength. This fascinating result opens up the possibility of making devices which exploit the wavenature of the electron, rather than its particle character. Consider the 1D channel not just as a quantum-scale *wire*, but as an *electron waveguide*. Now we can think of making quasi-optical or quasi-microwave devices but using electrons. For example, we could use interferometers and stub-tuners as switches, and directional couplers as a form of transistor.

A number of groups have tried successfully to make such interference devices, but the modulation of resistance, instead of varying from ON to OFF, is very small. The reason for this can probably also be learned from microwave engineering. A good interferometer has to be monomode, ie only one confined mode should participate in the action of the device. The electron waveguide can, it seems, only be made monomode if it is extremely short - a *point contact*. Any attempt to lengthen the point contact into a quantum wire waveguide loses the monomode character. The waveguide has to be made wider, and consequently more than one lateral mode participates in the transport. Why is this?

The solution may lie in some work carried out by the NANSDEV consortium. Theoretical studies of quantum wires showed that the potential in the region where the wire forms is not perfectly smooth. On the contrary, even though the donor ions in the

HEMT layer are remote from the wire, they nevertheless contribute a random potential within which the electron has to move. As the wire is squeezed down to monomode dimensions (around 40nm in width for a typical HEMT layer) the roughness in the potential becomes so severe that the wire breaks up into a series of disconnected puddles of electrons. In a very short contact, there is a good chance that a severe

fluctuation will be avoided and the channel conduct down to single mode widths, but in an elongated wire there is a high probability of encountering a fluctuation which cuts off a monomode channel. Consequently, extended wires have to be wider than desired in order to conduct. Figure (4) shows the calculated electron densities in point contacts without fluctuations (a), with fluctuations in a short contact (b) and (d), and in a longer contact (c).

Some experimental backup for these calculations was obtained by the NANSDEV group using a point contact which could be moved electrostatically in such a manner that the local potential 'landscape' could be 'mapped'. The results of this experiment confirm the existence of the fluctuations, although the calculations appear to overestimate their magnitude.

This model also explains why lateral quantum effects can only be detected at low temperatures. Because

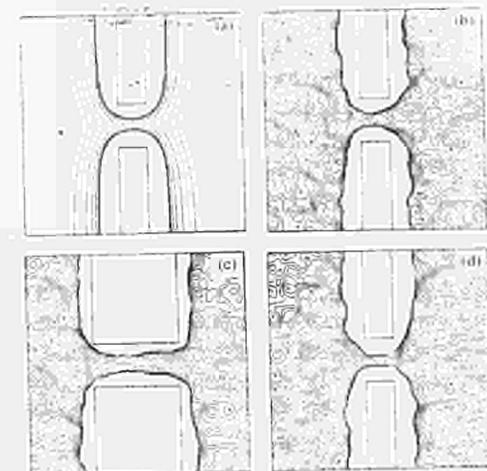


Figure (4). Calculate electron densities in a 1.5μm square around narrow point contacts. The gap between the gates is 0.3μm in all cases. A smooth potential and one with randomness are contrasted in (a) and (b), both for 0.2μm long gates. Figure (c) shows a 0.6μm long gate which (d) shows a 0.2μm gate with a well in the channel produced by the random potential

the wires are wide, the separation between the lateral subbands is very small. Unless random thermal energies are small compared with this separation, the electron will be able to switch from one lateral mode to another, smearing out any quantum effects. To improve the thermal robustness of the effects, the wire must be narrowed by further squeezing until the separation between lateral subbands is much greater than kT at the temperature in question. But the squeezing process runs into difficulties because of potential fluctuations, and so the operating temperature is limited to the cryogenic range.

The result sounds extremely pessimistic for the future of lateral quantum transport. But there are other ways to put electrons in quantum wires and better methods of defining the wire in the first place, both routes offering the possibility of greater immunity to fluctuations. In the case of wire definition, both the NANSDEV and LATMIC consortia are active, the former exploring the fabrication of quantum wires by damaging the high mobility layers outwith the wire using low energy ion beams, and the latter trying to intermix the semiconductor alloys on either side of the wire by the use of impurity induced disorder (IID). This technique first implants the layers with impurities and then anneals the structure for a short period. The presence of the impurities accelerates the

interdiffusion of aluminium into the underlying pure GaAs layer, giving rise to a confinement potential. Both techniques are in the early stages of development.

If a 1D wire can be thought of as an electron waveguide, what is analogous to a structure which is confined in all three dimensions - a 0D structure? A quasi-atom, of course. The electrons in such a structure - often referred to as a 'Quantum Dot' - will occupy the very particle-in-a-box states that most solid state physicists and engineers learn how to calculate at university. We can make such structures by confining electrons between two pairs of quantum point contacts, as the NANSDEV group have done, or take the route developed by the LATMIC group of making a double barrier (resonant tunneling) layer structure and etching it into narrow pillars. In the latter case it is possible to observe the fine structure in the I-V characteristics of the device attributable to electrons tunneling through the atom-like 0D states.

In an extension of this work, the NANSDEV consortium were able to take an array of these quasi-atoms and connect them together to form an artificial lattice. Again, the characteristics of this structure were exactly in line with expectations for a short array of atoms. Of course, the effects were only observable at very low temperatures but the result is rather important because it means that we are beginning to be able to fabricate structures in which vital parameters such as band gap energy are under the control of the device designer. Many applications exist for this type of structure, including the generation and detection of far infrared or sub-mm wavelength radiation. Such applications would require large arrays of quantum dots connected together to form the artificial lattice. Experiments underway in the NANSDEV consortium are designed to help us understand the properties of electrons in these arrays when illuminated by far infrared radiation. It has proved possible to detect the modifications to the infrared response which are characteristic of the lattice and work is in progress to attempt to understand the photoconductive behaviour of the device. Again, however, better means of confining the electrons will be necessary to enhance the effects which are brought about by the periodic array.

Recently, both consortia have been able to detect a remarkable effect in quantum dot systems: the presence or absence of a single electron. In very small semiconductor dots, the capacitance of the dot is itself extremely small. Should an electron enter the dot, the local change in potential due to that one electron is considerable. Since the energy levels inside the dot consist of a series of discrete atom-like laterally confined states, the lower of which will be filled, the entry of a single electron may be sufficient to prevent further electrons entering the dot, because the states into which another electron would have to enter are all occupied. Until an electron leaves the dot, no further electrons can get in. By using external ac signals on the gates forming the entry and exit barriers to the dot, the NANSDEV consortium have been able to clock individual electrons through the device. Consider the ramifications of this result in the future: might we see computers in which one electron represents one bit of information? It will take a long time to achieve that ambition: much smaller capacitors will have to be fabricated for this Coulomb Blockade effect to be detectable at room temperature. But in the shorter term we can contemplate applications of single-electron devices for extremely sensitive charge detectors and for current standards, where the current is defined by the electronic charge and the frequency of the ac signal used to clock electrons through the circuit.

4. Optical applications of laterally confined structures

Zero-dimensional 'quantum box' states have been detected by transport experiments, but it was the possibility of a quantum box laser which first excited the interest of the device community. I have already described the advantageous properties of a quantum well laser in terms of a narrow emission spectrum and high gain. A gas laser, on the other hand, is much more monochromatic but has the disadvantage of enormous bulk and great fragility. Suppose one could fabricate a laser having the narrow linewidths of a gas laser, but all the advantages of a solid state device. Quantum dots may offer such advantages. The device would emit light as a result of transitions between discrete laterally quantised states, and there would be no emission at higher or lower energies, because these states would not be occupied. Both the LATMIC and NANSDEV groups have been working to understand the fabrication and optical properties of these devices. A typical fabrication route involves starting with a GaAs/AlGaAs quantum well layer, which is then fabricated into quantum dots by a combination of electron beam lithography and reactive ion etching.

Figure (5) is a micrograph of a typical etched dot array fabricated by the NANSDEV consortium. Surprisingly, these devices luminesced strongly when illuminated with laser light. It had been expected that the dry etch damage caused to the sidewalls during fabrication would have destroyed the light emitting properties of the structure. Indeed, in some cases such loss of emission was detectable, but by annealing the samples in an arsenic atmosphere the light luminescence recovered strongly. Clearly the behaviour

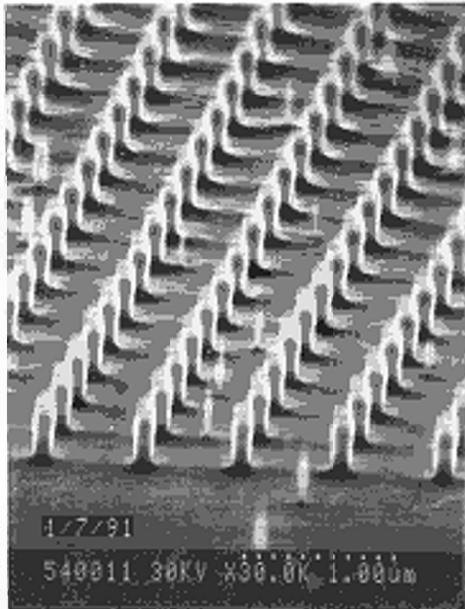


Figure 5. Array of quantum dots fabricated by dry etching of a quantum well layer. The dots in this array are approximately 100nm in diameter. The electron beam resist is clearly visible on the tops of the dots.

of these samples is very sensitive to the detailed history of the sample during fabrication. The LATMIC consortium has studied similar structures before and after overgrowth with AlGaAs. They observe an increase in the carrier lifetime after overgrowth and a corresponding improvement in luminescence output. LATMIC has also explored the use of the impurity induced disordering process referred to earlier with good results, and the advantage that no new surfaces are created during fabrication, thus avoiding contamination and damage.

One of the goals of current optical studies of quantum dot and wires is a proper understanding of the luminescence output. There is no hope of making a good laser device if the dots do not luminesce efficiently, yet there is considerable debate as to which factors control luminescence output. Both the LATMIC and NANSDEV consortia have evidence that there could be an intrinsic barrier to the emission of light in quantum dot and wire structures. When an electron is

created in a quantum wire or dot by laser stimulation or by the injection of a current, it has high energy and needs to relax down to the energy of the lowest subband before recombining to emit light efficiently. In bulk material the electron relaxes through a continuum of intermediate energy states without any difficulty. In quantum wires and dots, however, there may not be enough intermediate states available for this process to occur rapidly. Instead, the electron may find other ways to lose its energy, and may not recombine to emit light. In corroboration of this model, members of the LATMIC consortium have shown that the electron temperature in their wire structures can be abnormally 'hot', and the NANSDEV group has been able to fit their data on luminescence output from wires and dots to a computer model which takes account of this impeded relaxation. Work is in progress in both groups and in other laboratories to test this model further. On the one hand we must hope that it is incorrect or that the problem can be circumvented, otherwise it is difficult to envisage useful light emitting devices based on quantum dots. The situation may be eased in quantum wires, however. On the other hand, one can envisage useful devices - nonlinear optical switches, for example - which could be improved by the incorporation of quantum dots if the model is correct! It is fortunate to be in a position where, irrespective of the physics experiments, potentially valuable applications should emerge.

5. nanostructure fabrication by direct growth

The experiments I have described so far have largely been carried out in devices fabricated by traditional means, i.e. by growth, followed by lithography and metallisation or etching. And it is clear from the outcome of these experiments that if devices based on quantum confinement effects, or single electron effects, are to be of industrial rather than laboratory interest, we will have to make devices which are smaller than is currently possible. Structures fabricated by electron beam nanolithography could never be commercially viable. It would take many years to write a whole wafer with devices having minimum features sizes of the order of 10nm, because e-beam lithography is a time-serial technique, each element of the pattern being written in succession. Other methods of fabrication have to be found which avoid the need for e-beam lithography and also, hopefully, avoid the need for 'dirty' fabrication steps like dry etching, which create exposed surfaces and cause damage to the material.

A group of techniques of current interest can be referred to as *direct growth*. The idea here is not to grow a series of layers and then modify them by lithography, but to control the growth process in such a way that quantum-scale structure emerge naturally. One way to do this is to start with a series of grooves on the wafer and then to overgrow the structure with a series of quantum wells. Because the kinetics of growth depends on surface orientation, the grown wells are not uniform in thickness.



Figure 6. MOCVD direct growth on substrate patterned with V-grooves by holographic lithography.

For example, Figure (6) shows a series of quantum wells grown by Metal-Organic Chemical Vapour Deposition (MOCVD) on a surface etched into V-grooves using a grating mask fabricated by holographic lithography. The bright stripes are GaAs, and the dark stripes are the AlGaAs barriers. At the bottom of

the V, the GaAs is thicker than elsewhere. This is because the growth rate at the bottom of the groove is greater than on its sloping walls. If one calculates the confining potential in this arrangement, one finds that the thicker region of GaAs forms a quantum wire.

Other studies are under way in both the NANSDEV and LATMIC consortia to understand growth on surfaces with textured 'clues' to control the growth. The NANSDEV consortium has developed a computer model which accurately predicts the shapes of structures that emerge when dots and wires are overgrown using Molecular Beam Epitaxy. They have also shown that by judicious choice of the direction of the initial texture, the overgrown material 'self organises' into a pyramidal structure whose facets are smoothed by the growth process. It would even seem that by choosing the correct crystal plane for growth, there may be no need for any pre-definition of surface clues: the chemical nature of the surface itself provides sites around which nanometre-scale structure forms naturally.

This are extremely significant results. The only lithography required in this direct growth process is the initial optical lithography to make the V-grooves. Optical lithography is quick, because all elements of the pattern are defined in parallel. The growth process is just as fast as it normally takes to grow a quantum well. Moreover, the wafer that results, which could be covered with quantum wires or quantum dots, could be supplied directly to a device manufacturer who could use it as a 'smart' substrate!

6. Conclusions

Laterally patterned nanoelectronic devices are in their infancy. The studies taking place under ESPRIT Basic Research are largely designed to understand the physical principles which govern the optical and electronic properties of the structures. Yet exciting developments have already emerged: a visionary could contemplate a future in which arrays of nanometre-scale structures store and process information using single electrons handed from one device to another by quantum waveguides. Information would be addressed into the system using photons emitted from low power quantum dot lasers. Most of the effects needed for such a system have been demonstrated in the last three years. It may take another thirty years of research and development before it becomes a reality. The direction which that research has to take in order to make the necessary progress is quite clear, though in some cases formidable. We need smaller structures with stronger electron confinement, and fabrication techniques with better resolution and speed: but we are already seeing ways in which these goals might be achieved. Nanoelectronics may be only one vision of the future, but it is a vision with a firm scientific base.

7. Acknowledgements

I thank the coordinators of LATMIC and NANOFET for their help with the preparation of this article, and the members of the NANSDEV, LATMIC and NANOFET consortia for the information supplied.

MICROELECTRONICS

TIPBASE Bipolar Advanced Silicon for Europe

A. Pruijboom
Technical Co-ordinator TIPBASE,
Philips Research Laboratory Eindhoven,
Building WAG,
P.O. Box 80.000
5600 JA Eindhoven
The Netherlands

1. INTRODUCTION

Within the project ESPRIT 2016 TIPBASE, high performance bipolar technology is developed. The aim of the project is to strengthen the European market position in the application areas of telecommunication and consumer electronics. Envisaged products, in these areas, require technology with improved speed, reduced power consumption, increased packing density and good analogue performance. The main partners in this project are Europe's most competent IC-manufacturers, Philips, Plessey, Siemens, SGS-Thomson and Telefunken. Furthermore, two associate contractors and seven subcontractors are participating in TIPBASE. The project covers a period of three years. Effective work started on February 1, 1989, the preceding year being a definition phase. At the end of the project (February 1, 1992), the main partners will deliver a company specific demonstrator, showing the successful integration of TIPBASE technology in their application areas [1]. In addition, each of the partners will produce common demonstrators, which will serve as a benchmark for the developed processes.

In this paper the technical progress, made within TIPBASE, will be reviewed. A comparison will be made between performance of TIPBASE technology and that of state-of-the-art technology. Furthermore, results of advanced device research will be presented.

2. CONSOLIDATED PROCESS FOR DEMONSTRATOR CIRCUITS

All partners have consolidated the process flow for the demonstrator circuits. This process flow is denoted "Version 1" or V1. Since the application areas of the different partners, and hence their specific demonstrators, require different process optimization, the V1 flows are not identical. However, to demonstrate the improved speed and reduced power consumption of their V1 technology, the partners have to deliver common demonstrators by the end of July, 1991. These common demonstrators are a static frequency divider (16:1) operating at 10 GHz and ECL-ring oscillators with a gate delay below 30 ps and speed power product below 25 fJ.

Although, the developed V1 processes of the partners are not identical, the co-operation within TIPBASE has resulted in a high degree of commonality in these processes. All partners will use self-aligned structures, to realize links of deep submicron dimensions, between the intrinsic and extrinsic base of the transistors, without using submicron lithography. Even more, they have all decided to use inside-spacer configurations, in which the emitter width is smaller than minimum lithography dimen-

sions. An example of such a structure is the so-called double-polysilicon-bipolar transistor, shown in Fig. 1.

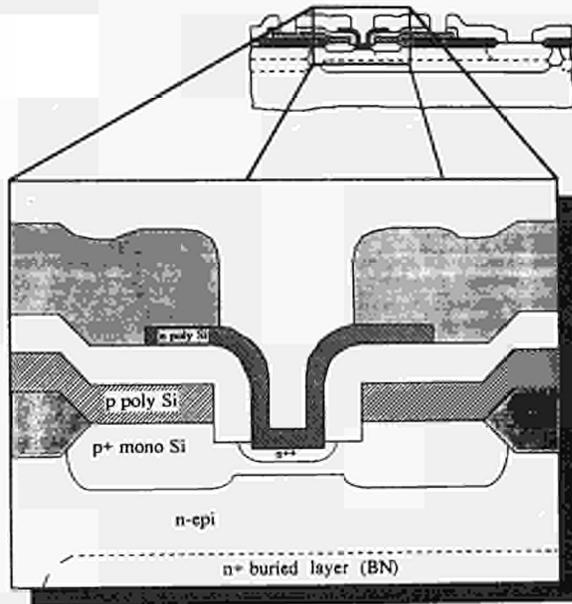


Figure 1: Schematic cross-section of self-aligned bipolar transistor, with inside spacers.

For all partners, base formation is realized using a shallow boron or boron difluoride implantation and the emitter is diffused from an arsenic-doped polysilicon layer. A cleaning process has been developed to obtain epitaxial layers with a defect density below 1 cm^{-2} . This is crucial for yield capability at 100k-transistor level. A buried-collector sheet resistance of $20 \text{ } \Omega\text{cm}$, in combination with an epi-layer thickness of $0.6 \text{ } \mu\text{m}$, has been achieved, using low-pressure and low-temperature epitaxy. These small epilayer thicknesses are essential, to meet the speed targets of V1. Two of the partners have decided to use trench isolation, for increased packing density. A common test mask, for multi-level metallization, has been designed, to be able to compare the reliability of the different metallization schemes of the partners.

The parameters of transistors, fabricated in V1 technology, have been determined. They are listed in Table 1 and compared with state-of-the-art technology. It should be noted that the parameters of the state-of-the-art technology are a collection of the best reported results and should, therefore, be interpreted with care. The speed and low-power performance of V1 transistors is at least a factor of 2 better than that of typical production technology. Also the complexity of V1, expressed by the metal pitch and the packing density, is superior to that of a typical production process. The R&D results are indeed better than those of V1. Especially the $f_T = 50 \text{ GHz}$ is outstanding. Firstly, it should be mentioned that the reproducibility of these results is unknown. More important, in the R&D results usually only one parameter is optimized. This is clear from the very low BV_{CE0} of 3 V, the high R_{pinch} and the fact that f_{max} , which is more relevant for circuit operation, is not reported. The listed circuit performance is comparable to results of advanced device research, which is performed in a separate work package in TIPBASE. These results will be presented in the next section. It can be concluded

that the consolidated process flow of V1, has a superior performance compared to state-of-the-art production technology and is well suited to reach the goals of TIPBASE.

Parameter	V1-process		typical production	R&D
	target	achieved		
f_T (GHz)	25	17 - 24	5 - 8	50
f_{max} (GHz)	-	22 (3V)	-	-
τ_D (ps)	30	34	70 - 250	24
$\tau_D P_V$ (fJ)	25	25	60 - 200	14
NF_{2GHz} (dB)	< 5	< 3	> 6	-
BV_{CE0} (V)	-	5 - 6	-	≈ 3
R_{pinch} k Ω /sq	10 - 20	11 - 17	-	17 - >50
$W_{E,min}$ (μ m)	0.5	≥ 0.3	1.2 - 2.0	0.2 - 0.4
metal pitch (μ m)	3.5	< 3	5	-
packing density (mm^2)	1000	-	200 - 300	-

Table 1: Cutoff frequency f_T , maximum oscillation frequency f_{max} , noise figure at 2 GHz NF_{2GHz} , gate delay τ_D , power-delay product $\tau_D P_V$, breakdown voltage BV_{CE0} , intrinsic base resistance R_{pinch} and minimum emitter width $W_{E,min}$, of V1-technology compared to parameters of state-of-the-art technology

3. ADVANCED DEVICE RESEARCH

A separate work package in TIPBASE is devoted to "New Devices". This work package serves to integrate basic research work into the project. This should allow for the evaluation of new technology and process trends and for the rapid transfer of such innovations into the process architecture. Inclusion of this advanced research activity provides an excellent chance to integrate novel research, of various European research institutions, into the project and to confront the research community with the actual needs of industry.

The work in this work package can be roughly divided into two parts. The first part deals with the investigation of new materials and new techniques. The second part deals with new device concepts. Part of these new device concepts are based on the previously mentioned new materials and techniques. Below some of the achievements are highlighted.

Heterojunction Bipolar Transistors with SiGe Base

With novel deposition techniques like molecular-beam epitaxy (MBE) or low temperature chemical-vapour-deposition (CVD), it is possible to grow epitaxially on a silicon substrate, strained layers of $Si_{1-x}Ge_x$. Due to the larger lattice constant of SiGe, the cubic lattice is tetragonally distorted. Depending on the Ge content, the bandgap ΔE_g in these layers is smaller than that of Si. For low Ge concentration, the bandgap narrowing amounts to about 8 mV per percent Ge. Doped layers of $Si_{1-x}Ge_x$ can be used as base layers in bipolar transistors such as to achieve a "heterojunction bipolar

transistor" (HBT). The collector current I_c , of such a HBT, is enhanced, with respect to that of an otherwise identical silicon-base transistor, and can be expressed as

$$\frac{(I_c)_{SiGe}}{(I_c)_{Si}} = C \exp\left(\Delta \frac{E_g}{kT}\right) \quad (1)$$

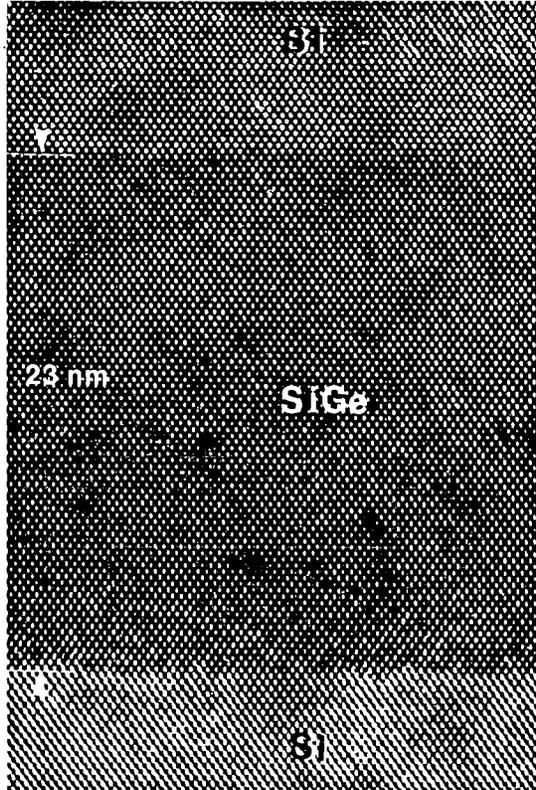


Figure 2: High resolution transmission electron micrograph of a cross-section of a 23 nm-thick $Si_{0.75}Ge_{0.25}$ layer, capped with Si.

Here, C is a constant of the order unity, which accounts for changes in densities of state. It is about 0.4, for $Si_{0.8}Ge_{0.2}$ [2]. Since the base current of the Si and SiGe-base transistors is the same, the current gain of the SiGe-base transistors is enhanced by the same factor as that of the collector current. Using the above mentioned values for $Si_{0.8}Ge_{0.2}$, this factor is about 200. The current-gain enhancement can be traded off against the base resistance of the transistors, giving the possibility to make transistors with very low base resistance while maintaining a sufficiently high current gain.

The incorporation of these layers in bipolar transistors imposes new demands on the device architecture. Furthermore, the strain in the layers can relax partially at higher temperatures (above ca. 850 °C), leaving misfit dislocations at the layer interfaces. Since these misfit dislocations should be avoided, the allowed thermal processing steps are restricted.

In TIPBASE, the growth of Si and SiGe layers with steep doping profiles is studied and devices are made, in which these layers are integrated. As an example of layer

doping, low temperature Si epitaxy has a large potential to improve the performance of bipolar transistors.

Base Layer	$R_{B,i}$ ($k\Omega/sq$)
500 Å Si 500 Å SiGe	3.8 4.5
1000 Å Si 1000 Å SiGe	1.7 2.0

Table 2: Intrinsic base resistance of bipolar devices with epitaxial bases of Si and $Si_{0.8}Ge_{0.2}$

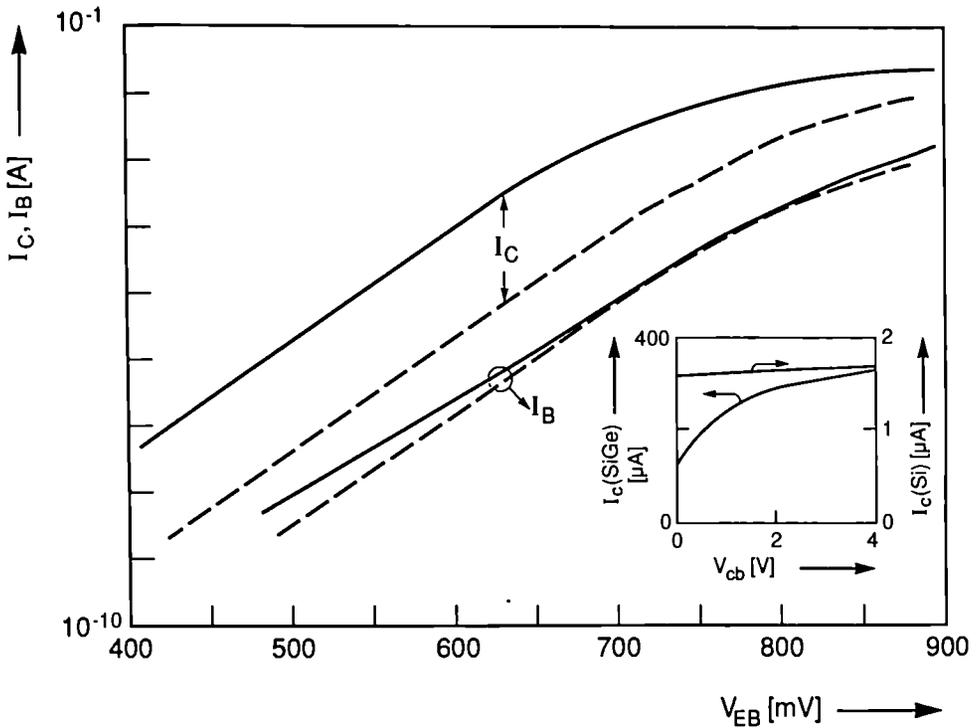


Figure 4: Gummel plot of a Si (dashed line) and $Si_{0.8}Ge_{0.2}$ (solid line) bipolar transistor with an emitter area of $12 \times 62 \mu m^2$. The inset displays the effect of V_{cb} on I_c for the Si and SiGe transistor at a fixed V_{eb} of 600 mV.

In Fig. 4, the collector and base current I_C and I_B of the 50 nm-base devices are given logarithmically as a function of the emitter-base voltage V_{EB} . Both I_C and I_B of the Si-base device display, over several decades of current, the ideally expected one decade increase per 60 mV V_{EB} . The SiGe HBT shows an ideal collector current and a small non-ideality in the base current. As is shown in the inset of Fig. 4, the collector-current enhancement in the SiGe device is, in agreement with our expectations, about 200, at

growth investigations, Fig. 2 shows a high resolution transmission electron micrograph of a cross-section of a 23 nm-thick $\text{Si}_{0.75}\text{Ge}_{0.25}$ layer, capped with Si.

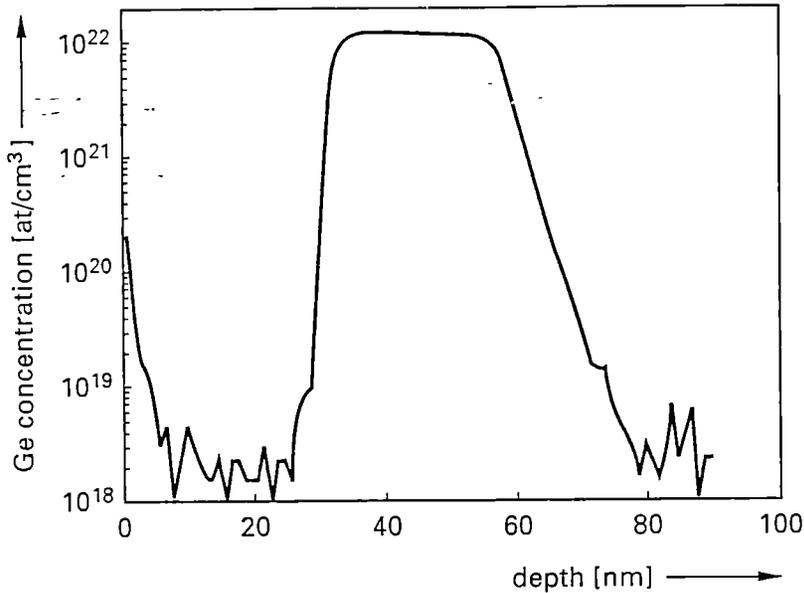


Figure 3: High resolution SIMS Ge profile of a 23 nm-thick $\text{Si}_{0.75}\text{Ge}_{0.25}$ layer, capped with 26 nm Si, grown at 625 °C.

The layer appears to be defect free with a sharp featureless interface. Fig. 3 displays the corresponding composition profile, as measured by secondary-ion-mass spectroscopy (SIMS). Both the leading and trailing edge, in this plot, are determined by the SIMS detection limit. These observations indicate that the interfaces are atomically sharp. The layer was grown at atmospheric pressure at 625 °C, in a commercially available Epsilon One reactor of ASM Epitaxy [3]. This is a very important result, because it brings the growth of strained SiGe layers closer to being a production technique.

MBE growth of Si and SiGe layers, at low temperatures, is also investigated, within TIPBASE. There is longer experience with this technique, but there is a general feeling that it is not suitable as a production technique, because of the limited throughput of MBE equipment.

However, since there is more experience with this technique, it can already be used to fabricate devices and assess the possible performance improvements in these devices. Devices have been fabricated, using MBE, with Si and $\text{Si}_{0.8}\text{Ge}_{0.2}$ -base layers. The devices were isolated using mesa etching, giving a simple device structure, that is also used by other researchers in this field, to characterize the properties of SiGe layers.

Table 2 lists the intrinsic base resistance of 50 and 100 nm-thick Si and SiGe-base layers doped by boron to a level of $2 \times 10^{18} \text{ cm}^{-2}$. Firstly, it should be noted that the values of the 100 nm-thick base are a factor of 8-10 lower than that of an implanted base with the same base width. This is due to the much steeper doping profile in the deposited base. This indicates that, in devices with deposited Si bases, the base width and hence the base-transit time can be reduced dramatically and that, even without Ge

a collector-base voltage of 2 V. These devices are the most ideal epitaxial-base devices that have been produced, so far, by MBE.

Selective Epitaxial Growth Transistors

Also part of the material research activities in TIPBASE is the selective epitaxial growth (SEG) of silicon layers, in oxide and nitride windows, by CVD techniques. These selectively deposited layers open new possibilities to realize transistors with further improved speed and power consumption. The drastically reduced temperature budget, after buried layer formation, allows for the fabrication of devices with extremely steep and shallow buried layer profiles. Therefore, for example, the electrically active collector depth can be scaled down considerably. Further possibilities of SEG-technology are the fabrication of sidewall-contacted base and sidewall-contacted collector configurations with reduced parasitics.

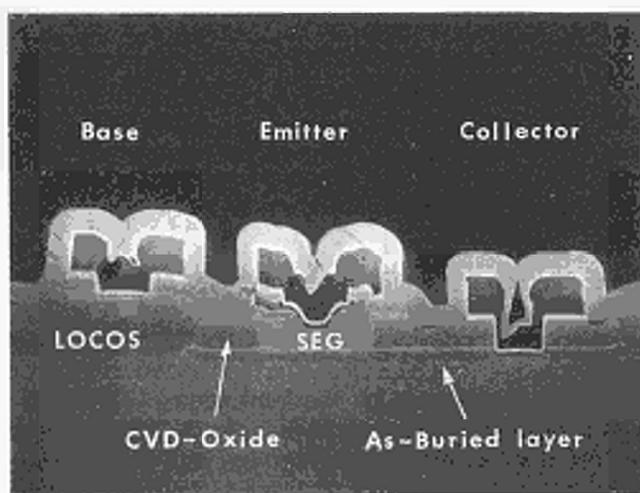


Figure 5: SEM cross-section of the SEG-transistor

Like the situation for SiGe layers, there is also an activity in TIPBASE to integrate SEG-layers in bipolar devices. Fig. 5 shows a scanning electron micrograph (SEM) of a cross-section of an SEG-transistor [4].

After buried collector formation a silicon dioxide layer is deposited, by CVD and patterned by reactive ion etching. Subsequently, a collector layer is selectively deposited. The emitter-base configuration was fabricated using the inside spacer structure that is described above, in the first paragraph. Measured SIMS doping profiles, of the active transistor region, are depicted in Fig. 6. The diffused arsenic buried layer is only 450 nm thick and exhibits an excellent transition sharpness of less than 50 nm over two decades of doping.

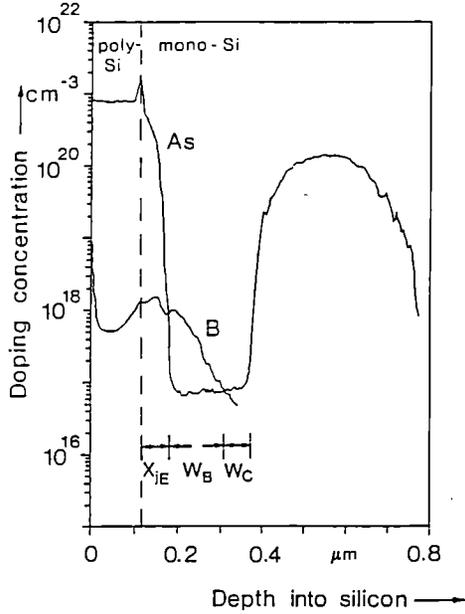


Figure 6: SIMS doping profile of the active area of the SEG-transistor

The frequency response for devices with $A_E = 0.8 \times 10 \mu\text{m}^2$ is given in Fig. 7, showing a maximum cutoff frequency f_T of 32.5 GHz. The high speed capability of the SEG-technology has also been investigated by the fabrication of CML-ring oscillators. The measured power-delay performance is shown in Fig. 8. Transistors with relaxed $1 \mu\text{m}$ design rules ($A_E = 0.8 \times 2.8 \mu\text{m}^2$) exhibit minimum gate delays of 32.6 ps at a power consumption of 2.9 mW/gate. Due to the excellent thickness uniformity of the SEG-process the standard deviation for the oscillation frequency is only about 2%. Gate delays as well as power consumption are significantly improved using transistors, designed with tight $0.8 \mu\text{m}$ design rules, with an effective emitter area of $0.35 \times 2.0 \mu\text{m}^2$. In this case, the resulting minimum gate delay is 26 ps at a power consumption as low as 1.3 mW/gate. The estimated value for the power delay product is about 12 fJ. These excellent results are comparable to the best reported R&D results, quoted in the first section.

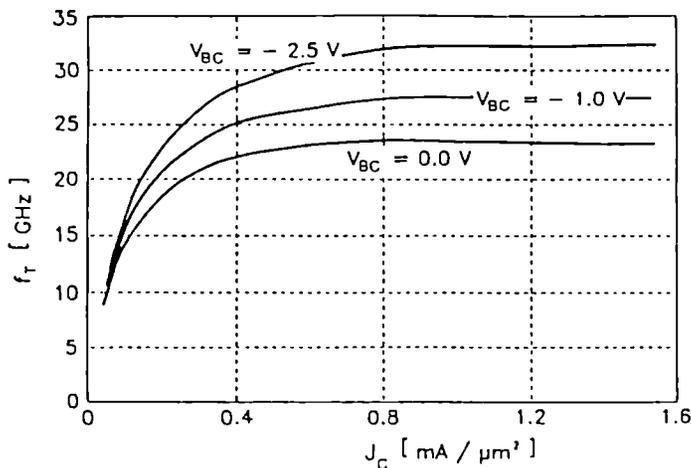


Figure 7: The cutoff frequency f_T of SEG-transistors with $A_E = 0.8 \times 10 \mu\text{m}^2$ as a function of the collector current

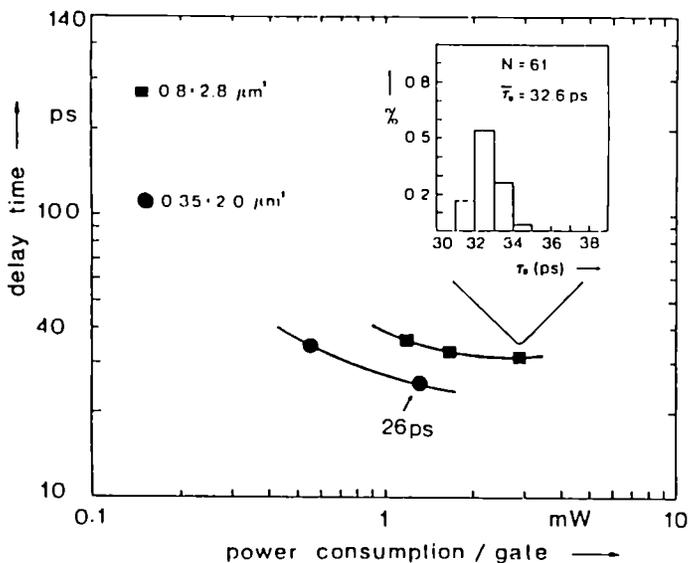


Figure 8: Power delay characteristics for relaxed $1 \mu\text{m}$ and tight $0.8 \mu\text{m}$ design rules, respectively.

4. CONCLUSIONS

In TIPBASE pre-production technology has been developed with a superior packing density and a speed and low-power performance, which is at least a factor of two better than typical production technology. This technology is required for envisaged applications in telecommunication and consumer electronics. Also advanced device research

is performed in TIPBASE. The results of this research are outstanding and comparable to the best R&D results reported in literature.

REFERENCES

- [1] A.J. LINSSEN, L. TREITINGER, R. KAISER, P. WARD, J. SMITH, Very high speed bipolar technology and circuits TIPBASE Bipolar Advanced Silicon for Europe, Esprit '90 Conference Proceedings, Kluwer Academic Publishers 1990.
- [2] A. PRUIJMBOOM, J.W. SLOTBOOM, D.J. GRAVESTIJN, C.W. FREDRIKSZ, A.A. VAN GORKUM, R.A. VAN DE HEUVEL, J.M.L. VAN ROOIJ-MULDER, G. STREUTKER, G.F.A. VAN DE WALLÉ, Heterojunction Bipolar Transistors Grown by Molecular Beam Epitaxy, IEEE Electron Device Lett., July 1991, accepted for publication.
- [3] W.B. DE BOER AND D.J. MEYER, Low-temperature chemical vapor deposition of epitaxial Si and SiGe layers at atmospheric pressure, Appl. Phys. Lett. 58, 1286 (1991)
- [4] T.F. MEISTER, H.W. MEUL, R. STREINGL, D. HARTWIG, R. WEYL, I. KERNER, R. MAHNKOPF, R. SCHREITER, J. WENG AND R. KPL, A 26 ps selective bipolar technology, 1991 Symposium on VLSI Technology, Digest of Technical Papers, 67 (1991)

ACHIEVEMENTS IN HIGH DENSITY PACKAGING

P. Chantraine, Bull, Les Clayes-s-Bois
N. Chandler, GEC-Marconi, Gt. Baddow
J. Brandenburger, Hoechst-CeramTec, Marktredwitz
J. Chilo, INPG-LEMO, Grenoble
Y. de Maquillé, MCTS, Mantes-la-Jolie
C.O'Mathuna, NMRC, Cork
W. Koschnick, Siemens-Nixdorf, Munich
R. Bargain, Souriau, Mechelen
R. Dümcke, Technische Universität Berlin.

SUMMARY

The APACHIP project is making excellent progress on Multi-Chip Modules and on material such as TAB tapes, connectors and ceramic-based packages, all suitable for high density, high pincount applications. Now entering its third year (of four), its contract has been extended following an independent review of progress and of the on-going work programme.

The work programme covers all areas from high density chip assembly and interconnection, using very fine substrates, to packaging and testing. Design is also included. Functional demonstrators have been designed and are being built, to show how the developments can be incorporated into exploitable products. They vary in size from a few ICs to many dozens, in speed and power (bipolar or CMOS) and in cooling method. There are also novel single-chip packages. The demonstrators indicate how widely these advanced interconnection and packaging techniques could be applied and make a significant industrial impact.

INTRODUCTION

The project started in 1989 and has progressed extremely well, both technically and in collaboration between the partners. The main goal remains the development of high performance interconnection and packaging technology for advanced VLSI circuits and the demonstration of its availability through deliverable demonstrators. These will be the basis from which marketable products can be generated with only limited additional development effort. These products will be not only packaged electronic assemblies (Multi-Chip Modules) but also packaging material supplies (such as TAB tapes, connectors and ceramic-based packages) necessary for an advanced European electronics industry and expertise in modelling and in reliability assessment.

The collaborators have organised the project into nine Technology Tasks and three groups of Demonstrator Tasks:

- T1 TAB Tape Technology
- T2 High Density Connection Technology
- T3 Ceramic Substrate Technology
- T4 Polyimide-copper Substrate Technology
- T5 High Density Organic Substrate Technology

T6 Cooling Technology
 T7 Electronic Modelling
 T8 Connector Technology
 T9 Reliability Assessment

D1 Single-Chip Package Demonstrators
 D2 Air-Cooled Demonstrators
 D3 Water-Cooled Demonstrators

The APACHIP project represents a very significant advancement in capabilities, being an improvement not only in miniaturisation but also in performance.

TAB TAPE TECHNOLOGY

The main goal for TAB tape during the first part of the project was to develop new materials, technologies and equipment to produce in an industrial environment significant quantities of TAB tape with 316 Input/Output (I/O) leads at a pitch of 125 μm , on 70 mm tape (see Figure 1). MCTS has successfully reached this target and today can provide industrial quantities of such fine TAB tape for the world market. In the near future, ICs will have pitches as fine as 100 μm and 75 μm , for which TAB appears the best assembly technology. The goal for the next two years is to develop new materials and technologies to produce tape with up to almost 600 leads on these finer pitches, on an industrial scale, for applications such as Data Processing, LCD displays and High Definition TV.

HIGH DENSITY CONNECTION TECHNOLOGY

During the first two years the emphasis has been on TAB assembly, at 125 μm pitch. Bull has successfully developed bonding equipment and optimised processes including lead-forming and bonding both inner and outer leads at 125 μm pitch, with a 12 mm IC on 70 mm tape, onto multi-chip substrates. This has involved close collaboration with other partners - NMRC for test chips, MCTS for tape and Siemens-Nixdorf for production engineering. Extensive practical work has led to optimising the bonding process, in particular focussing on single-point ILB to avoid difficulties with gang-bonding such large ICs. Also the bonding head principle and construction have been modified to eliminate the risk of silicon cratering. The process is under control for both ILB and OLB and multi-chip demonstrators have been successfully assembled. New developments in the next two years will address TAB at 100 μm and 75 μm pitch.

High density wire-bonding has recently been added, to take advantage of the latest developments in two ways. The first is as an alternative to TAB, for process flexibility and short lead-time. Wire-bonding on tape could provide parallel solutions to TAB. Secondly, modified wire-bonders could replace the complex bumping process conventionally used to prepare wafers for TAB. This new "ball-bumping" process could be applied directly to the standard IC metallisation. Targets include up to 600 I/Os at 100 μm pitch.

CERAMIC SUBSTRATE TECHNOLOGY

The use of ceramic packages (single or multi-chip) is increasing for highly integrated ICs, since packaging and thermal management are key issues. Hoechst CeramTec has

improved the fabrication technology in various respects (e.g. shrinkage control, via-hole size, fine pattern definition) and has developed different types of package: a single-chip package with multilayer metallisation; a Pin Grid Array (PGA) with improved heat transfer (using a W-Cu insert); another PGA with inner lead pitch of $200\ \mu\text{m}$; a multi-chip package substrate for subsequent polyimide/copper deposition, 10 cm square with 888 connector pads (see Figure 2). In the next two years a single-chip package with 448 leads on 0.015" ($380\ \mu\text{m}$) pitch is planned. For multi-chip modules the ceramic shrinkage tolerance will be further reduced ($< 0.5\%$).

POLYIMIDE-COPPER SUBSTRATE TECHNOLOGY

The aim of this task is the development of processes to make a stack of dielectric and conductive thin films, a polyimide-copper multilayer, as the basis for Multi-Chip Modules (MCM-D). The increase of I/O count, at reduced pitch, and the electrical performance require a high connectivity, leading to fine signal lines and small via-holes. In the first two years of APACHIP, Bull has established the different process steps for such a multilayer structure, with up to 5 metal layers, on top of a 10 cm square co-fired ceramic substrate. A pair of signal layers is sandwiched between two power/ground layers, the pitch of tracks and via-holes both being $100\ \mu\text{m}$ gives a connectivity of $200\ \text{cm}^2/\text{cm}^2$ (see Figure 3). Surface tracks at $125\ \mu\text{m}$ pitch have tin-lead finish for TAB of ICs. Though these processes have been established with ceramic substrates they can also be applied to silicon substrates.

A functional module is being made with the above features, on a ceramic base. In years three and four, process development will continue for quality and yield and will be used for final test vehicles. In parallel the pitch of tracks and via-holes will be reduced to $80\ \mu\text{m}$.

HIGH DENSITY ORGANIC SUBSTRATE TECHNOLOGY

The first two years of the project have seen the successful development of the technological steps to realise a new level of high density organic substrates for Multi-Chip Modules (MCM-L). The basic processing steps, in many ways similar to Printed Circuit Board techniques, are essentially established and the transfer into production has already been started. Technology demonstrators have been designed and manufactured which show: conductor tracks at pitches down to $80\ \mu\text{m}$ (and thicknesses up to $30\ \mu\text{m}$ for low resistance) suitable for TAB or for wire-bonding (see Figure 4); via holes down to $30\ \mu\text{m}$ diameter; controlled impedance tracks; up to 17 metal layers (signal and power/ground). A novel method of bare-substrate testing has also been demonstrated, using gas-discharges as contacts. Presently at $635\ \mu\text{m}$ pitch, this will be enhanced in years 3 and 4 by reducing the pitch to $100\ \mu\text{m}$ and enlarging the area to $200 \times 200\ \text{mm}$.

By combining the capabilities of Siemens-Nixdorf and GEC-Marconi it will be possible to achieve further rapid progress to come to an early exploitation. For mainframe computers, it is intended to develop a large substrate (25 cm square) for a demonstrator within APACHIP. The design parameters have been derived from experience in the first two years and in line with new developments in IC integration, which are placing even higher demands on the substrate than previously envisaged. The aim is to build MCMs which comprise complete mainframe processor units on a single substrate, with features so fine that fewer layers are needed.

Smaller MCMs can already use the technology in its present state, giving size reductions by typically a factor of six compared to dense Surface Mount Technology, as well as improved performance. Applications include computers and processors for IT, telecommunications, aerospace, automotive, industrial and consumer products.

COOLING TECHNOLOGY

The aim is to develop highly efficient cooling for Multi-Chip Modules. To provide for a wide range of power dissipations, several different approaches are being developed. In the first two years, efforts have concentrated on the basic thermal modelling and the realisation of prototype structures for the different cooling techniques: direct contact conduction (with water cooling) (Siemens-Nixdorf); liquid immersion (GEC-Marconi); heat pipe with flexible adhesive (Bull). Direct contact between the ICs and a cold-plate is highly efficient, resulting from high pressure, excellent surface conditions and optimum heat transfer to the cooling medium. An ultrasonic (non-destructive) test procedure has been developed to assess the chip/cold-plate contact, a preliminary approach for the correlation between ultrasonics signals and thermal resistance has been established. The thermal resistance has been dramatically decreased by coating the cold-plate with soft metal.

Devices dissipating at high power densities can be cooled very effectively by immersion in a perfluorinated liquid, the device temperature being controlled by selecting the liquid's boiling point. Heat removal is by convection and conduction at low power densities and by nucleate boiling at higher densities. This approach obviates the need for precision engineering of a cold-plate. A thermally conductive membrane separating the coolant from the electronics allows water to be used, offering even higher levels of heat extraction.

For lower heat dissipations, a heatpipe/flexible adhesive cooling system will lead to sufficient thermal contact between chip and cold-plate.

All the above cooling techniques and the most important packaging parameters have been experimentally compared, in strong relationship with the Functional Demonstrator requirements, in the form of hardware modules. A common test chip has been developed by NMRC for this work and assembled to the various substrates using TAB.

Employing the results from other tasks, the work in years 3 and 4 will concentrate on improved test vehicles which will be used to indicate how the different thermal management techniques could be implemented in practice i.e. in products.

ELECTRICAL MODELLING

For the electrical modelling, the first step was to define the propagating impedances of the various signal tracks used in the single-chip packages and in the various MCM substrates. Investigations by LEMO have mainly concerned the analysis of inductance, capacitance and characteristic impedance variations versus changes in geometrical parameters such as conductor and dielectric dimensions and dielectric constants. Coupling coefficients and cross-talk between lines have also been investigated. Theoretical results derived using geometrical data from the partners have been compared with experimental data obtained by LEMO on test vehicles designed and built by the partners. A theoretical model has been developed for signal lines over meshed ground planes. Results show a significant increase in line impedance and a strong increase in coupling between lines, as more copper is removed from the mesh.

Connectors being developed by Souriau have also been modelled, giving good correlation between the model and measured results.

For years 3 and 4 the Technical University of Berlin has joined the project to simulate and measure interconnection tracks for high speed applications. A major goal is the development of a software tool for simulating lossy transmission lines in high density MCMs, which aids the design of controlled impedance interconnect with controlled cross-talk. Simulations of 2 and 3 dimensional structures will help to develop layout design rules and to reduce the need for cost-intensive prototyping. Simulations will be verified using test structures by measuring signal transmission properties.

CONNECTOR TECHNOLOGY

To provide a new generation of high speed, high density connectors for MCMs, the electrical modelling has been used as the basis for the design. A concept has been derived for Surface Mount signal contacts with low electrical length and for ground contacts. To achieve the high density, the fabrication of very thin insulator walls has been mastered. Prototypes have been designed and made having 888 contacts arranged around the four edges of a substrate, at 0.635 mm pitch (see Figure 5).

In the next two years, results from the first prototypes will be analysed and taken into account in defining final prototypes. The concepts will also be improved to prepare the prototypes for industrial exploitation.

RELIABILITY ASSESSMENT

The use of novel materials and an increasing number of process steps for fabricating MCMs poses significant reliability issues. Scanning Acoustic Microscopy (SAM) has been evaluated by NMRC as a means of analysing multilayer interconnect media and to define an optimised SAM system for this application. Subsequent work focussed primarily on developing quantitative analysis techniques for measuring material properties and on evaluating SAM for detecting defects in multilayer structures, including correlating SAM, X-ray and microsection images of die-attach and hermetic package lid seals. Specially optimised acoustic lenses have been designed and fabricated, to analyse TAB bonds in year 3. Quantitative analysis of adhesion in multilayer structures will be investigated in year 4.

NMRC has also modified the design of an existing test chip, as a result of discussions with APACHIP partners. It consists of arrays of heat sources (up to 50 W) and diode temperature sensors, spread across the 12 mm chip. A daisy-chain allows evaluation of interconnect continuity and bond resistance. These test chips were used for optimising TAB ILB and OLB processes and for evaluating the cooling techniques. Further test chips have been assembled onto demonstrator substrates and will be used in years 3 and 4 to characterise the thermal performance of the cooling systems and of the single- and multi-chip demonstrators.

SINGLE-CHIP PACKAGE DEMONSTRATORS

Single-chip package construction during the first two years was an evolution of existing configurations (at Bull) to improve understanding of the electrical behaviour. Typical characteristics are: 276 I/Os (plus eight voltage connections) on 0.0125" (318 μm) pitch; cavity to accept a 12 mm IC and package foot-print less than 30 mm square (see Figure 6). The main aims have been to define a structure compatible with existing

test equipments, to enhance electrical performance especially during simultaneous switching of buffers and to capitalise on other developments within the project. Associated activities are electrical modelling, IC on high density TAB frame and the inclusion of low inductance built-in capacitors. The assembly processes (die attach, TAB and lead-forming) have been optimised, parts have been successfully assembled and electrically tested, showing performance at least as good as complex ceramic multilayer packages. Quality evaluation has revealed no degradation of the packages after thermal shock and pressure-cooler ageing. Thus high performance has been demonstrated in a package that could be used in future products at lower cost than conventional ceramic packages. This represents a transition towards Chip On Board (COB) technology which is becoming ever more popular and which may be applied to high performance systems. For the remainder of the project a new 448 I/O package has been defined, for which high density wire-bonding will be used.

AIR-COOLED DEMONSTRATORS

The purpose of the air-cooled demonstrators is to show the feasibility of integrating up to sixteen CMOS ICs, each 12 mm square, on a 10 cm square substrate, with a better electrical performance than conventional mounting on a PCB. Two different substrate technologies are being compared: firstly the polyimide-copper on ceramic (Bull and Hoechst-CeramTec), secondly the high density organic substrate (Siemens-Nixdorf). The co-fired ceramic substrate includes ground and voltage layers and signal vias to transmit signals between the top and bottom layers. The sixteen ICs are assembled by TAB, capacitors are soldered on both surfaces. The two substrates are compatible at the connector level, each fitting into the new connector developed within the project by Souriau, and thence attached to a PCB. In normal use the MCMs will be hermetically sealed. Micro-bellows will transmit heat from the ICs to the package lid, whose fins will be air-cooled at 2 msec^{-1} to extract 50 W. Electrical tests on both types of module will be performed in the coming months. A final demonstrator will be developed in years 3 and 4 to test the ability of the assembly to transmit still higher frequency signals and to finalise the thermal and mechanical construction of the module.

WATER-COOLED DEMONSTRATORS

The objective is to develop packaging and cooling systems for high power (bipolar) MCMs. Thermal modelling has been followed by successful development of several functioning demonstrators. The first incorporates the Siemens-Nixdorf high density organic substrate with TAB mounted ICs. A mechanical trial module has been assembled in the Siemens-Nixdorf production plant, as planned for production (see Figure 7). A fully functional test module will be assembled in the coming months and evaluated thermally and electrically. The second demonstrator is similar to the first but uses instead the polyimide-copper on ceramic substrate, to compare the mounting and cooling behaviour. Several mechanical modules have been built and optical methods have been introduced to evaluate contact between the ICs and the cold plate (Newton interference rings). The results show that both types of substrate can be used in the Siemens-Nixdorf style of module, with sufficient mechanical/thermal contact between ICs and cold plate.

As a result of continuing developments in ECL ASICs and the requirements for larger MCMs for large data processing systems, another Functional Demonstrator has been

designed and all the important mechanical parts have been specified. A high density organic substrate, 25 cm square, will carry up to 80 ECL ASICs. Increasing silicon integration leads to much higher power dissipation (around 5 kW per module), so an improved water-cooling system is needed to ensure reliable heat extraction while keeping the ICs below 60° C. The soft metallic layers between ICs and cold plate will optimise heat transfer. The main activities in year 3 will concentrate on thermal modelling, test substrate design, thermal test chips and a test bed for the complete module. During year 4, the thermal and electrical behaviour will be measured.

MEMBERS OF THE ESPRIT 2075 (APACHIP) CONSORTIUM

Organisation	Principal Contact	Telephone No.	Telefax No.
Bull (Leader) (nr Paris)	K. Kurzweil	33(1).30807048	33(1)30807833
GEC-Marconi (nr Chelmsford)	N. Chandler	44 245 73331	44 245 75244
Hoechst CeramTec (Marktredwitz)	J. Brandenburger	49 9231 69236	49 9231 62409
INPG-LEMO (Grenoble)	J. Chilo	33 76 87 69 76	33 76 43 37 96
MCTS (nr Paris)	Y. de Maquillé	33(1)34768630	33(1)34769322
NMRC (Cork)	C. O'Mathuna	353 21 276871	353 21 270271
Siemens-Nixdorf (Munich)	R. Hillemann	49 89 63647390	49 89 636 41840
Souriau (Mechelen)	R. Bargain	32 15 41 49 61	32 15 41 58 29
T.U. Berlin (Berlin)	R. Dümcke	49 30 31423553	49 30 313 6733

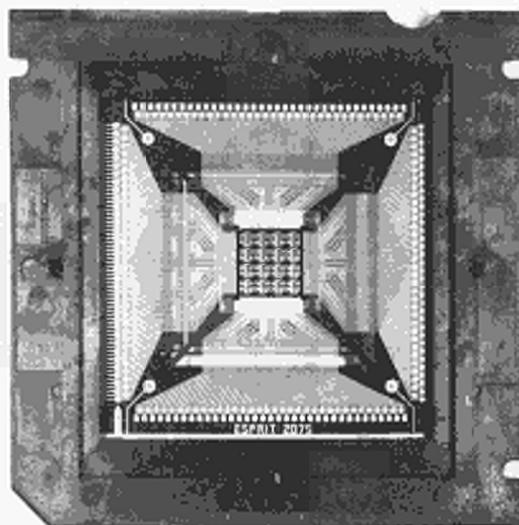


Figure 1 Thermal test chip, 12 mm square, on 70 mm TAB tape, ILB pitch 125 μ m, lead count is 276.

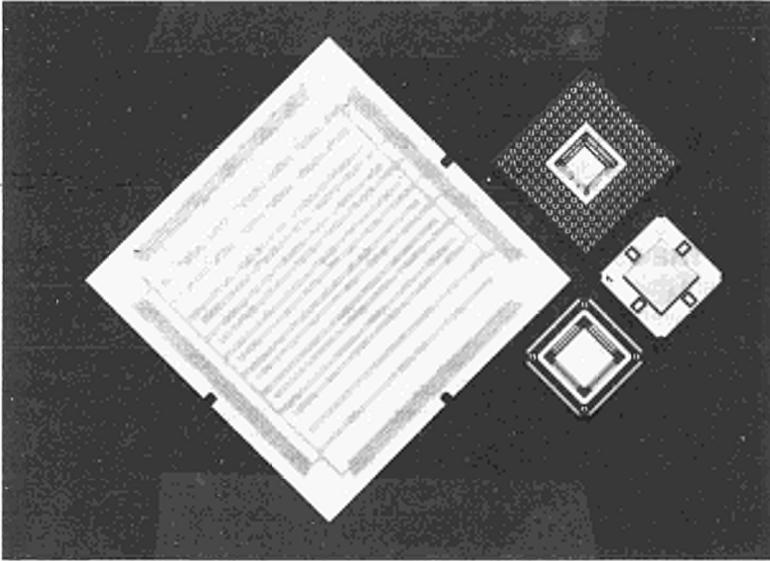


Figure 2 Co-fired ceramic substrate for Multi-Chip Module, 10 cm square having 888 pads to match the high density (0.635 mm pitch) high speed connector. Also shown are single-chip packages (Pin Grid Array and Surface Mount styles).

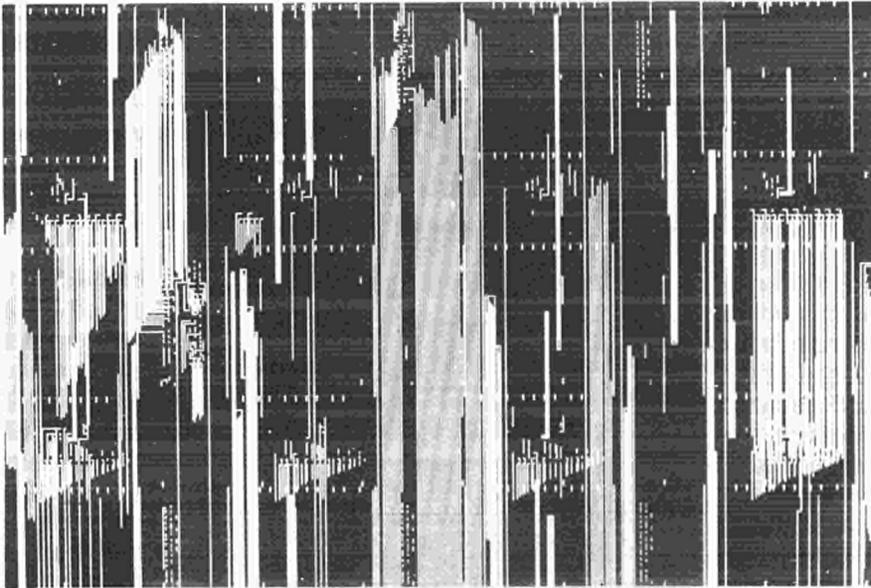


Figure 3 Details of third metal layer in polyimide/copper substrate, track and via-hole pitch 100 μm .

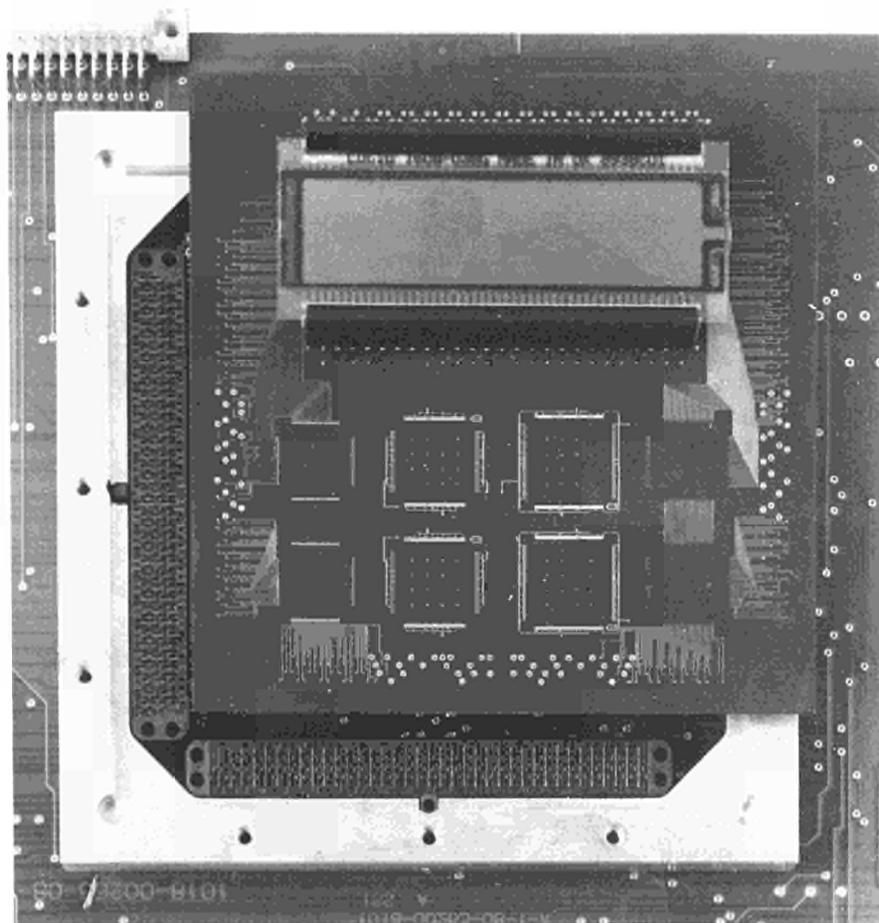


Figure 4 High density organic substrate featuring tracks at pitches down to $80\ \mu\text{m}$ and laser-cut-via-holes down to $30\ \mu\text{m}$ diameter. Devices will be assembled on both sides using TAB and SMT packages.

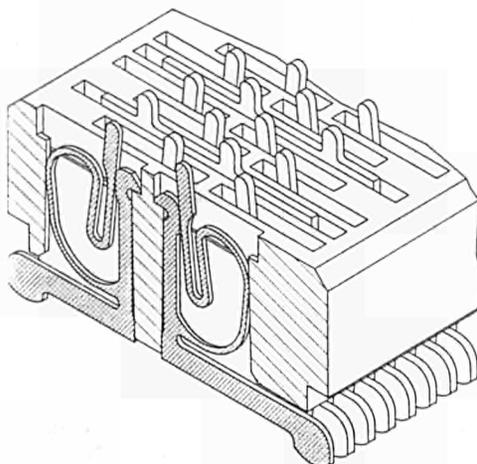


Figure 5 A new generation of high speed, high density connectors for MCMs has low inductance, controlled impedance contacts at $0.635\ \text{mm}$ pitch.

SCP1

APACHIP PROJECT

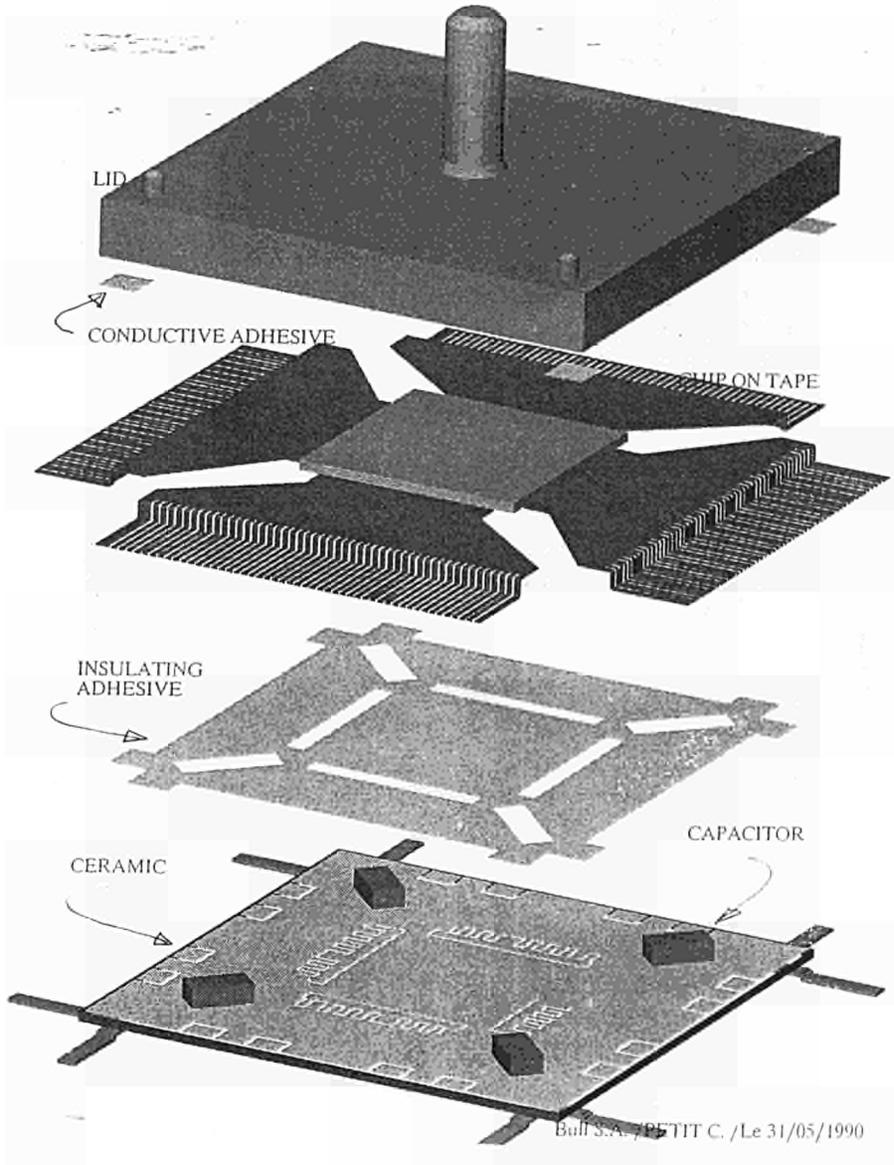


Figure 6 Single-chip ceramic packages, using 12 mm chip-on-tape, 284 I/Os having 0.0125" (318 μm) pitch at outer lead frame.

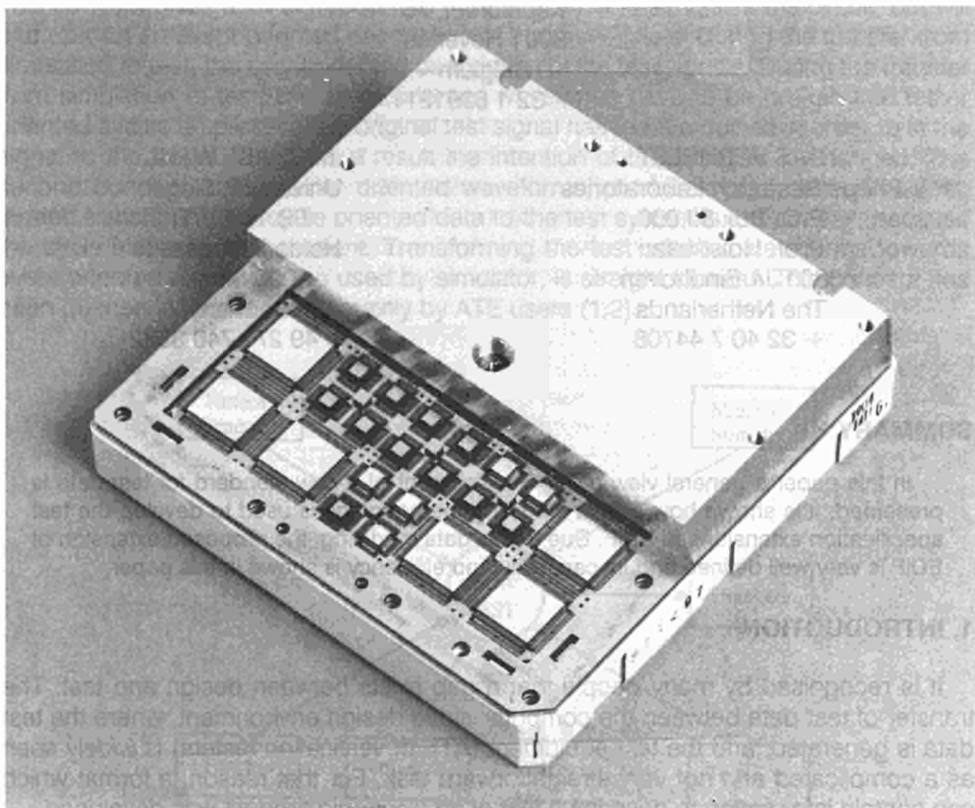


Figure 7 Multi-Chip Module using a high density organic substrate, with 16 VLSI and 32 LSI devices mounted by flip-TAB. The VLSICs are 12 mm square and have 316 I/Os.

EDIF AS A STANDARD TEST SPECIFICATION FORMAT

PAUL VANDELOO, IMEC
Kapeldreef, 75
3001 Heverlee
Belgium
+ 32 1 6281211

BAS VERHELST
Philips Research Laboratories
P.O. Box 80.000
Prof.-Holst-Laan
5600 JA Eindhoven
The Netherlands
+ 32 40 7 44708

MICHAEL WAHL
Universität Siegen
FB 12 DV/TI
Holderlinstrasse 3
5900 Siegen
Germany
+ 49 271 740 3332

SUMMARY

In this paper a general view of the development of a new standard for test data is presented. It is shown how a data modeling methodology is used to develop the test specification extensions of EDIF. Due to this data modeling, the proposed extension of EDIF is very well defined and its capability and efficiency is shown in this paper.

1. INTRODUCTION

It is recognised by many people that a gap exists between design and test. The transfer of test data between the computer aided design environment, where the test data is generated, and the test equipment (ATE or verification testers) is widely seen as a complicated and not very straightforward task. For that reason, a format which can contain information to describe a test of an electronic product was developed.

The Electronic Design Interchange Format (EDIF) is a well accepted standard for certain applications. The last years, considerable effort is spent by the EDIF Technical Test Subcommittee to extend this format for testing. This paper gives a general overview of the latest developments with respect to the new EDIF standard for test. The paper does not go into the syntactical details of EDIF, but the extensions are explained to show the performance of the new standard.

2. PRINCIPLES AND CONCEPTS OF TEST DATA INTERCHANGE

The new proposal of a standard for test (EDIF extensions) addresses the exchange of data between different environments. The concepts that have played a role in the definition of the standard for exchange of information are linked to the Test Specification Concept as presented in (1). In this proposal, data can be exchanged between software CAD tools, between CAD tools and test equipment, between testers themselves and between testers and CAD tools (figure 1). The data which can be interchanged in the context of this new standard is more than the transfer of ones and zeros across an

interface. It can include the information for diagnostics, shorts testing, schmoo testing where the parameters are varied and the length of the test may be extended.

When the design and test environment have to be integrated, two approaches can be followed: the use of independent simulator data and the use of tester oriented waveforms. Using the concept of independent simulations, the designer will optimize and choose the signal timing in such a way that his requirements are met optimally. This concept gives a lot of freedom to the designer who simulates the circuit, but will end up with an event oriented description of the test signals. During the transfer from simulation to test, the event oriented description of the test signals. During the transfer from simulation to test, the event oriented waveforms have to be broken into tester oriented cycles. In practice the original test signal has to be modified in order to fit the signal to the target tester. As a result the intention of the designer is changed. The second concept, using tester oriented waveforms, produces much less problems. Indeed transferring the cycle oriented data to the test system is rather easy, because the cycle information is present. Transforming the test oriented waveforms towards event oriented waveforms as used by simulator, is straight forward. This concept has been pushed in recent years, mainly by ATE users (1,2).

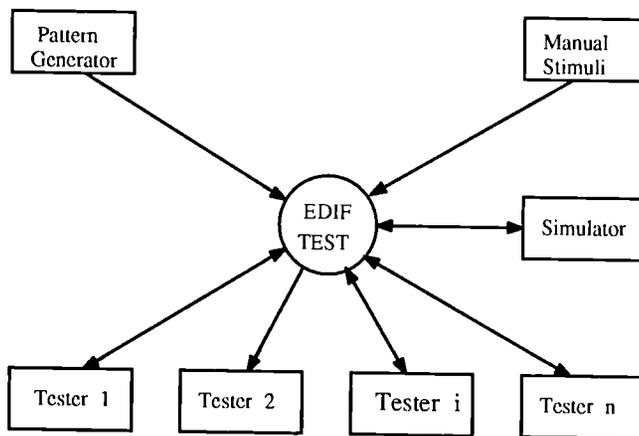


Figure 1: Data Interchange

The EDIF Test Extensions in its actual form are in principle developed to exchange tester oriented waveforms. This choice was made because of the growing importance of the representation and the ease to transform the information to the tester. The simulation or event representation is considered to be important for designers and has therefore to be stored in another view as for instance a dedicated simulator view. EDIF can be extended to support event oriented, time linear simulator waveforms in a following phase.

3. HISTORY AND CURRENT USE OF THE EDIF TEST EXTENSIONS

Standards usually arise from a company's internal agreement on interfaces and data transfer files. To get the agreement accepted by a wider audience, i.e. more than one company, a standardisation body has to approve the standard. Shortly, the EDIF

Technical Committee will approve a standard for test and will support this standard in the new release.

The activities of the European EDIF Test Technical Subcommittee (formed by European universities and electronic companies) have started in 1987. The committee's main interest was the implementation of test patterns and timing. This resulted in a set of preliminary documents, describing syntax and semantics. In 1988, the American EDIF Test Technical Subcommittee produced similar documents. It was seen that in order to integrate the "Extensions for Functional Testing" in one global test view, conceptual and data modeling techniques were necessary to be able to grasp the complexity of the problem (3).

At the same time the ESPRIT EVEREST project was defined. When this project started in 1989, a user requirement specification (4,5) (a wish list of future users) was generated. It appeared that beside a format which is able to represent test patterns and timing, many second priority topics need support in the near future. These include electrical information, parametric tests, test flow, fault dictionaries and others. Recently, boundary scan is added to the list. The members of EVEREST, working on the definition of a test specification format, preferred to extend an existing standard which is already in use a design specification format. There were two candidates: WAWES (which is a format based on VHDL(6), and EDIF(7). Conceptually there exists a great difference between VHDL and EDIF. VHDL can be considered as being a language for representing circuit behaviour algorithmically, whereas EDIF is optimized to describe the structure of a device. Looking at the purpose of the EVEREST project with respect to a test specification format and the requirements of the future users, EDIF was considered to be more appropriate.

When the EVEREST project decided to cooperate with the EDIF Technical Test Subcommittee, the EDIF Technical Committee announced EDIF 3.0.0, for which proposals and extensions could be submitted. The committee asked for a data model and the syntax is considered of less importance. The necessity of having a model is outlined in (3). Different members of the EVEREST project started a systematic approach to standardize a test specification format. In the first instance, a conceptual model was developed. This conceptual model outlines what has to be described and which relations between the information exist. The conceptual model has then to be refined. The entities and relations are defined by word, first, and then implemented using a formal language. The final result is the information model. This model has great advantages. First of all, the data is described clearly and formal methods can be used to check the validity of the model. In addition, the model gives the great advantage that it can be mapped into other standards. This can be seen for instance in the STEP initiative (8), where an overall product description standard is developed, which is described as an information model.

At the end of 1990, EVEREST project members together with the EDIF Technical Test Subcommittee have finalized an information model. After the complete definition of this information model, the syntax can be derived easily from the model. Next, the model and a proposal for the syntax were sent to the EDIF Technical Committee and at the time of writing this paper, we are waiting for the final approval.

Due to the support through the EVEREST project software based on the EDIF data model could be developed. First, a tool was implemented which allows to automatically generate an EDIF-like syntax as well as a procedural interface (PI) directly from the data model. Based on the PI, interfaces to a Philips internal format and to two testers, Tektronix LV 500 and HP 82000 were implemented. The interfaces proved the feasibility of the EDIF test extension and are in use.

4. THE CONCEPTUAL MODEL

Over the years it was found that it was not wise to define a standard by discussing syntactical details of that standard. As an alternative, a more systematic approach was followed by using modeling techniques. The learning curve of modeling is not easy to take. However, it is worth the effort since the coherence and consistency of the now standard test specification format depend essentially on the model. The modeling is done in two phases: the conceptual model and the information model.

The conceptual model is the start point of the discussions. It seems to be very difficult to find a consensus on the components constituting a test. After long sessions of brainstorming and debates, the basic model as shown in figure 2 and first presented in (9) was accepted. Although the step of conceptual modeling is very difficult and many people want to continue the standardisation work without a consensus on that model, the discussion is relevant and should be continued. Indeed the conceptual model will determine the content of the information model and will thus outline what the standard can describe.

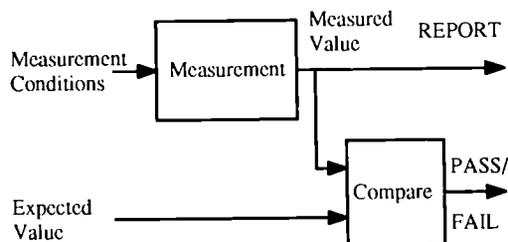


Figure 2: Basic Model of a Test

In the basic model of a test, as shown in figure 2, the data is distinguished from the operations performed on the data. The operations which are performed in the test are the measurement and the comparison. These two operations can be identified in every test procedure, although in a pure digital test the measurement and comparison are usually performed by the hardware and therefore hardly visible for the user.

The measurement is set up by stimulus conditions and measurement conditions. Stimulus conditions are given by electrical and environmental signals and must be applied to the DUT (analogue or digital electrical quantities, temperature, humidity, etc.,.....). For synchronous digital applications, the signals are formed by vectors, timing frames, cycle lengths and voltage levels. Some or all the outputs are observed and the output data is acquired. Response conditions, applied to the measurement instruments determine where and how the responses of the DUT are to be measured, e.g. strobe point, strobe window, loading,.....The result of the measurement can be logged into a test report or can be compared with a set of expected values. An example of the latter is the comparison of the digital vector response with the expected response vectors as occurs in a functional test.

5. THE INFORMATION MODEL

The second step in the modeling of a test specification format is the definition of the information model. The information model is based upon the conceptual model in terms of objects (e.g. test, measurement, frame, logic value.....) and relations between

objects. The objects are called entities and the connections between the entities are called relationships. Hierarchic relationships among entities are described by super-type/subtype trees. Note that information modeling should be carefully distinguished from database modeling (for instance the entity relationship model). The information model - in its purest form - defines and handles the information itself, not the physical realisation. However, it should be mentioned that the information model since aspects of test were taken into account. It was felt that in the area of test the efficiency of the data structures are a key issue with respect to usability and acceptance of the standard.

Contrary to the arbitrary notation on conceptual models, there is a clear directive from the EDIF Technical Committee to use the formal information modelling language EXPRESS (9) as a textual means of description. EXPRESS sets the focus on the definition of information classes, called entity types. Entity types are defined in terms of attributes which, in turn, are presented by other entity types. Entities are considered to be independent, but constraints can be given to effect dependency. This is done by rules in the global context of a where clause in the local scope of an entity type. As an example consider the following description of some entities in EXPRESS:

Example 1:

```
ENTITY edge
    name    : OPTIONAL name_def
    placement: time_point;
    state   : logic_state;
UNIQUE
    name;
END_ENTITY;
```

This example shows the description of an edge. An edge defines when a transition can occur. An edge is thus a specification of a time point and a logic state. This is written down straight forward in EXPRESS. An edge is also a named entity, which means that an edge can be referenced (for relative timing for instance), but the name is optional.

Example 2:

```
ENTITY logic_state
    SUPERTYPE OF (logic_constant XOR logic_from_table XOR logic_mapping_of;
END_ENTITY;
```

A logic state is either a constant and known logic value, or a value which is read from a table of logic values or a mapping of a logic value. The latter means that the next logic state depends on the actual logic state value and a transition table (a typical example is the surrounded by complement format).

Example 3:

```
ENTITY logic_from_table
    SUBTYPE OF (logic_state);
    possible_values : EXTERNAL SET [1:#] OF logic_value;
    serial_number   : OPTIONAL INTEGER;
WHERE
    (* serial_number is present if there are multiple logic_from_table's
    serial_number + 1;
END_ENTITY;
```

In this last example is shown how a logic table is built up. First of all, the model indicates that `logic_from_table` is a subtype of `logic_state`. This of course no new information with respect to the information of example 2, where it was stated that `logic_state` is a supertype of `logic_from_table`. The sub-super type relations are typical for the information model and are used to define the hierarchy.

Although the EXPRESS language can be read by humans, it is much easier to parse the input by a computer program. By doing so, it is possible to check the semantic consistency of the model. Presently, only syntax checkers are available, but it is expected that other tools will be available soon. EXPRESS also offers the possibility to generate graphical representations of the textual form of the model. This subset of the EXPRESS language is called EXPRESS_G. Programs for creating graphics from the textual EXPRESS representation are under development. Figure 3 illustrates the use of EXPRESS-G.

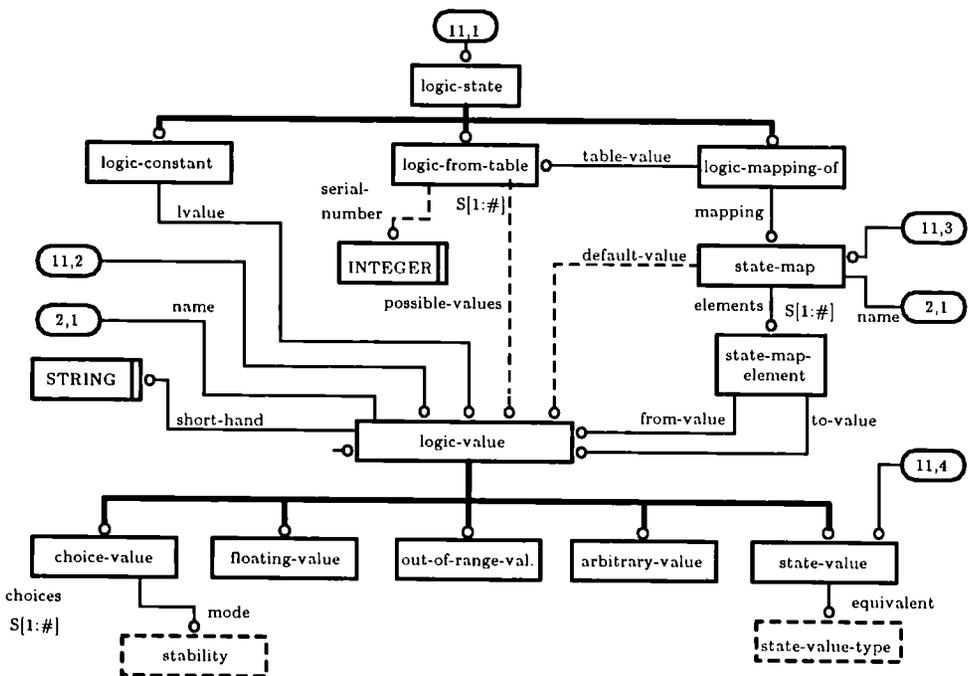


Figure 3: Example of EXPRESS_G Representation

6. THE BASIC MODEL OF A TEST

This section describes the information model for test. Due to the size of the model, it is not possible to present the model as a whole. Some examples will show the capabilities of the EDIF Test Extensions. The model as a whole is explained in (10,11).

The test plan is the outer wrapper of the "test view". It is the top level of the data model tree. A test plan or test program contains tests. The execution order of those tests is undefined. The `test_plan` entity contains a definition of tests as well as administrative data, scaling factors for units, a description of the Device-Under-Test

(DTU), and definitions of some entities that can be referenced by name. Data not referenced in any test is never used and thus ignored.

A test plan is the top-most entity. In EDIF, it is the test view. The EXPRESS representation of test_plan is as follows:

```

ENTITY test_plan
  name                : name_def;
  name_of_design      : STRING;
  administration      : OPTIONAL status;
  units                : OPTIONAL scaling;
  dut                 : dut_description;

  logic_values        : OPTIONAL VISIBLE SET [1: #] OF logic_value;
  state_values        : OPTIONAL VISIBLE SET [1: #] OF state_map;
  frames              : OPTIONAL VISIBLE SET [1: #] OF frame;
  levels              : OPTIONAL VISIBLE SET [1: #] OF level_group;
  p_tables            : OPTIONAL VISIBLE SET [1: #] OF pattern_table;
  f_tables            : OPTIONAL VISIBLE SET [1: #] OF frame_table;
  l_tables            : OPTIONAL VISIBLE SET [1: #] OF level_table;
  c_tables            : OPTIONAL VISIBLE SET [1: #] OF cycle_table;
  wv_timings          : OPTIONAL VISIBLE SET [1: #] OF waveform_timing;
  wt_tables           : OPTIONAL VISIBLE SET [1: #] OF wf_timing_table;
  elval_tables        : OPTIONAL VISIBLE SET [1: #] OF el_value_table;
  signals             : OPTIONAL VISIBLE SET [1: #] OF test_signal;
  tests               : OPTIONAL VISIBLE SET [1: #] OF test;
END_ENTITY

```

In the test plan, all the information which is global for a test is described. Several entities of test plan contain information which can be referenced in other parts of the model. The test_plan itself has also a name, which is used to discriminate it from other test_plans that might exist for the same DUT. In EDIF this is the name of the test view. The name of the designs is kept in name_of_design. This name is not used inside this data model. The administration information is also present in an EDIF file to identify the status (data origin, creation time, etc.) of that EDIF file.

The units provide a scaling for units (time, voltage, current). Voltage scaling serves as a constant multiplication factor for voltages. Each voltage value must be to identify the status (data origin, creation time, etc.) of that EDIF file.

The units provide a scaling for units (time, voltage, current). Voltage scaling serves as a constant multiplication factor for voltages. Each voltage value must be to obtain the physical value in Volts.

The DUT description gives characteristics (now only ports) for the Device-Under-Test. This information would be stored in the interface section of an EDIF file. It should be mentioned here that all the signals and timing are defined at the terminal pins of the device under test (and not at the testhead pin of the tester as was done in other standards). As a consequence these pins must be defined and therefore the dut_description is not optional.

Logic values, state maps, frames and levels are used in digital signals (quad-tables). The quad tables consist of pattern tables, frame tables, level tables and cycle tables. For power supply ports and analog ports, waveforms can be constructed. For this purpose, wavwform-timings, waveform_timing tables and electrical value tables can be defined. Both quad tables and waveforms are test signals, the definition of which is made here.

Finally tests are defined. They describe the actual test of the DUT by referring to the data defined previously. The sequence in which the test are executed is undefined. It

is at the level of test_plan that future extensions for testing can be added easily. For instance in the future the order in which the tests have to be executed will be defined in an entity flow. An extension for fault dictionary is under preparation and will be added under test_plan.

Let us now analyse the information model and find a method to describe a synchronous digital signal. In the information model, this signal can be represented by four tables. First of all, the total signal is divided into distinct time intervals, called cycles. During each cycle and for each port, four information items are available:

- a logic value, which holds the digital information to be applied to the port
- a frame, which determines at which time during the cycle the logic value has to be applied
- a cycle duration, which determines at which time during which the cycle is active
- electrical levels, which specify the mapping of logic values in the electrical domain.

When combined, these four items yield a voltage function versus time for many ports. The four information items are put into tables. The four tables can be found under the entity test_plan.

The first table is the frame_table, which specifies the timing frames per port and per time unit. These determine the time offset at which the logic values of the pattern array must be applied or expected from the DUT. The frame table contains names to reference a frame which has to be active in each cycle (change-on-the-fly). Such a name can be NRZ, RC, SBC,The frames as NRZ, RC, SBC,.....are not built in, but have to be defined beforehand in the definition section. In the frames construct under testplan, the different frames are defined. Every frame needs a name, since it has to be referenced later on. A frame is built up of so called edges, which describe a signal during the cycle. The edges specify a time point at which the signal changes its logic value and can specify a transition to:

- a fixed logic value
- a value from a table of logic values
- a mapping of a logic value from a (small) conversion table of logic values.

In case of an output port, the compare window is given by stating that outside the window an unknown logic value is expected, while inside the window the logic value is expected, indicated by the data in the logic values table. The model allows to make distinction between unspecified values, arbitrary values, choice values and out-of-range values. It is very important to understand that the frame table contains reference to frames, which are defined under test_plan.

The second table is the level_table in which the voltages corresponding to logic values are referenced. The voltages themselves have to be defined previously in the definition section. Remember that these definitions can be found under test_plan. In the levels construct, a voltage map specifies the voltage.

9. Conclusion

In summary, it can be stated that a lot of effort is spent in order to develop a new standard for test. This standard is based on an information model and can be used for the most important data exchange in the test environment. The approach used leads to a standard which is well structured and easy to understand. In addition, the information model can be used for the evaluation of other standards and is essential for the integration of standards in the future.

REFERENCES

- (1) Verhelst, Bas (1989). Using a Test Specification Format in Automatic Test Pattern generation. Proceedings of first European Test Conference, Paris.
- (2) Becu, J.: (1989). UTILE System : a Unified Environment from Simulation to Test. Proceedings of First European Test Conference, Paris.
- (3) Chalmers, D., Meys F. (1989). Successful CAD Integration needs a Standard Conceptual Model. Proceedings of European EDIF Forum, Bonn.
- (4) Evaluation of Existing Standards (1989). EVEREST WP2 internal report PHI 0074 CR RS.
- (5) User Requirement Specifications for TSF (1989). EVEREST WP2 internal PH1 0073 CR RS.
- (6) WAVES Language Reference Manual (1990). Waves Analysis and Standardisation Group of the Design Augmentation Standards Subcommittee (DASS) and the Standards Coordinating Committee Number 20 of the IEE.
- (7) EDIF Electronic Design Interchange Format, Version 2.0.0, Second edition (1990). Electronic Industries Association, EDIF Steering Committee, Washington.
- (8) Schroder, K., Ungerer M. (1989). Approach to Harmonization of EDIF and STEP. Proceedings of European EDIF Forum, Bonn,.
- (9) Schenk, D. (1990) EXPRESS language Reference Manual. Mc Donnell Aircraft Company, St. Louis
- (10) EVEREST Work Package Two Tutorial (January 23th 1991). Les Clayes sous Bois
- (11) EDIF Test Tutorial during the European Test Conference 1991, Munich.

Documentation is available, contact : Michael Wahl, Chairman EDIF Test Technical Subcommittee, European Chapter, Universitat Siegen, FB 12 DV/TI, Holderlinstrasse 3, D 5900 Siegen 1

The work presented has been partly funded by the ESPRIT project 2318: European Vanguard Efforts on Research and Engineering of systems for Testing (EVEREST).

N/A RESEARCH INTO BOUNDARY SCAN TEST IMPLEMENTATION

Sylvain KRITTER and Tiana RAHAGA
SGS THOMSON MICROELECTRONICS
Central R&D - CAD and Integrated systems
GRENOBLE - FRANCE

SUMMARY

This paper describes two BIST elements that have been designed using an Hierarchical Self-test Concept in an IEEE 1149.1 (JTAG) environment. At chip level, a general self-test structure to implement the JTAG RUNBIST instruction has been chosen and the implementation of a macrocell to test embedded static RAM following this architecture is described. At the board level, a chip to test Static RAM board array is presented. This chip is fully JTAG compatible, and supports the RUNBIST instruction with the methodology previously described.

1. INTRODUCTION

Testing is a concern for several people. IC designers include test features to test chips after manufacturing, and these are used on boards where test strategies are implemented by system designers. Moreover, there is a large demand for diagnosis of faults at chip as well as at board levels. The management of complexity, and the need for fast turn around are becoming dominant today, for both the system designer and the IC designer. A solution to this problem can be an hierarchical and standardized approach, where everyone benefits from the work done by others.

Bist

The complexity of the systems requires to partition them and to apply known technics for each element, according to their specific architecture and fault set. Built-In-Self-Test has the benefit of a fast execution time and limits the number of test-patterns to store, but we can see that this technic is not very common in the industrial world. One reason for that is the difficulty to know precisely the fault coverage related to physical failures, though theoretical works are done in this areas and figures for several cases can be approximated or measured; another is the practical difficulty in putting together different known technics at different levels (chip, board, system), and make them work and talk together.

Hierarchy and Standard

A better efficiency can be obtained if the testing problem is tackled globally, this is what is attempted in the Hierarchical Self-Test Concept (1), where each test feature can be controlled by its upper hierarchical level.

In this concept, hierarchy makes use of international standards to normalize the interfaces between the levels. At the system level, the MTM bus (Module Test & Maintenance Bus, IEEE P1149.5, not yet standardized) is a candidate to make the links between boards and the maintenance processor. The JTAG bus (IEEE 1149.1) is the

communication support between chips, at board level (figure 1), it also provides the requirements for running internal self-test within the chips, with the RUNBIST instruction.

IC and Board BIST elements

This article describes the elements we have implemented at the silicon and board level, around the JTAG protocol, to evaluate the feasibility and costs of such an hierarchical technic. A test scheme at chip level has been chosen : the Built-In-Self-Test

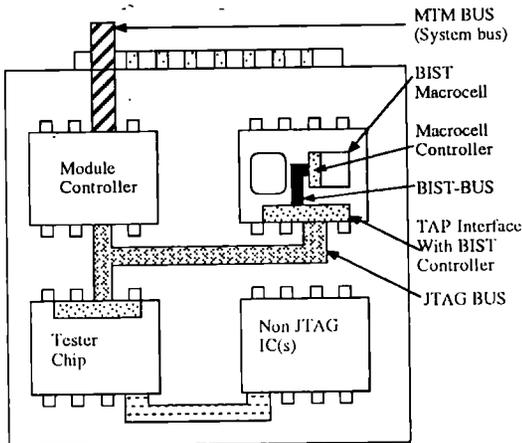


Figure 1. Hierarchical Self-Test Structure

and Boundary Scan Test (BIST-BST) method that has been developed in (1), (2). This method is based on a self-test architecture with an internal test bus, the BIST-BUS; it describes also the test sequences and the way to activate them, to be able to support the JTAG RUNBIST instruction.

Two BIST building blocks are described : a macrocell to test embedded static RAM (SRAM BIST MACROCELL), and an IC dedicated to test Static RAM board arrays (SRAM TESTER CHIPS).

The SRAM BIST Macrocell is driven by the BIST-BUS and sticks

to the test architecture required by the BIST-BST method; a diagnosis mode has been added to the test modes to determine which part of the SRAM array is defective.

The SRAM TESTER CHIP is entirely driven by the JTAG BUS, is activated through JTAG instructions and is self-testing conforming to the BIST-BST method. This chip is intended to become a catalogue product.

Gain

This work will allow to extract data from real elements, as well at physical level (area, speed) as at concept level (applicability), for chips and systems.

2. BIST AND BST METHOD

A method and a self-test architecture have been chosen to implement the RUNBIST instruction at IC level. This method is based on the work done by Siemens; more details can be found in (1). This method is not intended to be universally applicable to every IC, but can be experimented with, for some architectures and functionalities, mainly those with full synchronous design and no drastic constraints on speed performances (general constraints for full scan designs). In the following, a practical implementation is proposed and experimental design data is extracted.

Self Test Structure

Let us consider the self-test structure proposed in (1) of a Boundary Scan Compatible chip. We suppose that the constituting elements of such a chip can be classified in the following way, according to the test method to be applied (figure 2).

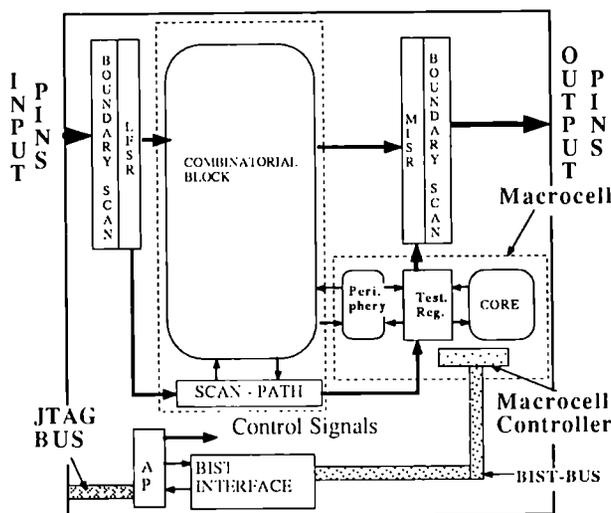


Figure 2. IC Self-Test Architecture

– Bist Macrocell.

A Bist macrocell is a cell where a test method exists and has been implemented; it can be a ROM, RAM, PLA, adder... Usually its test method is related to the specific structure of the macro (logical function, regularity) and is made to detect either functional faults (coupling between memory cells for a RAM...) or structural faults (classical stuck-at).

The Bist macrocell is constituted of:

- The Core. This is the functional part of the macrocell (for example

a complete RAM with sense amplifier, decoder...).

- The Test Register. It makes a clocked interface between the Core and the rest of the chip, and can be configured as scan-path. It has self-test functions during the Core test sequence, and holds the self-test result. This register may have also functional purpose, or be specially added for test purposes, in this case a Boundary-Scan-like architecture can be used.
- The Macrocell Controller. This controller is activated through the BIST-BUS, it runs the test sequence for the Core, and controls the functional modes of the Test Register.
- The Periphery. This is the link between the macrocell and the rest of the circuit (buses, buffers..).
- Random Logic. This is the part of the chip where no specific test method other than random testing can be applied in a BIST scheme, either because no regular structure can be exhibited, or because its size doesn't require a specific block to be implemented (small PLA or RAM for example). This block can be split into Combinatorial block and Scan-Path block, and will be tested by Random Vectors Stimuli and Signature Analysis.
- The Jtag blocks.
- Boundary Scan Register. The cells attached to input pads will be used also as a Centralized Test Pattern Generator, and those attached to output pads as a Centralized Signature Analyzer. The Test Pattern Generator and the Signature Analyzer are configured at the same time in parallel for the parallel links with the combinatorial blocks, and in series for the scan-paths.
- TAP and BIST INTERFACE. A BIST interface (BISTI) is added to the JTAG TAP-INTERFACE. This block controls the BIST macrocells through the BIST-BUS and the

scan-paths modes; it also generates the different test sequences for all the test resources.

Self-Test procedure

This architecture is devoted to activate the self-test of the different parts of the circuit following the RUNBIST protocol defined in the IEEE 1149.1 standard. According to this standard, the self-test should proceed autonomously: all the BIST sequences have to be launched without external actions. The BISTI block is in charge of activating the different test resources in the correct order. Two main phases are used:

- Test of the macrocells. All the BIST macrocells are put in self-test mode by the BIST-BUS. Afterwards, they run in parallel their self-test function. When completed, the signatures held in their Test Registers are serially sent to the Centralized Signature Analyser.
- Test of the combinatorial block. The scan-paths linked to it (including those of the BIST macrocells) are used to apply random patterns. The result of these stimuli is captured in these scan-paths and is serially shifted towards the Centralized Signature Analyzer. During this phase, the links between the BIST macrocells and the other circuits on the chip are also tested.

Others test sequences can be generated by the BISTI block to test reset or set pins, if any, or to enable several clock schemes, according to the specific architecture of the chip.

3. A SRAM BIST MACROCELL

Here is described a SRAM BIST Macrocell following the Hierarchical Self-Test concept previously described. Diagnostic capabilities have been added in the BIST macrocell, this feature also uses the BIST-BUS, without interfering with the other functions. The BIST-BUS and the different test modes associated to it are detailed for this example. Several implementations of a dedicated block for testing embedded static RAM have already been presented (3), (4), (5); for each case, the hardware used is specific to the application and is difficult to reuse in another environment.

Cell Number	INIT	S1	S2	S3	IS4
1	IR ₀ ↑↑↑	IR ₁ ↓R ₀ ↑	R ₁ ↓↑↓	R ₀ ↑R ₁ ↓	
...
k	R ₀ ↑↑↑	R ₁ ↓R ₀ ↑	R ₁ ↓↑↓	R ₀ ↑R ₁ ↓	
...
N1	R ₀ ↑↑↑	R ₁ ↓R ₀ ↑	IR ₁ ↓↑↓	IR ₀ ↑R ₁ ↓	

R0 : Read Data Background 0
 R1 : Read Data Background 0
 ↑: Write Data Background 1
 ↓: write Data Background 0

General description

The macrocell performs a test of embedded Static RAM according to the Marinescu Algorithm (17N) (6) (table 1), without necessary knowledge of the row/column implementation. This algorithm is run with different Data-Backgrounds (7) according to the width of the memorized words, to detect bit coupling within a word.

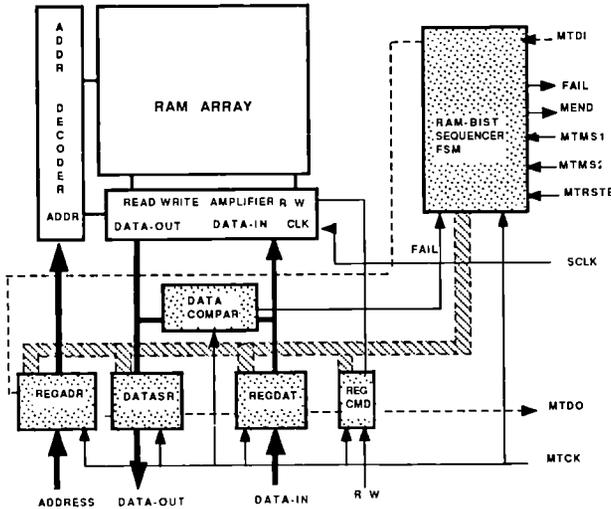


Figure 3. SRAM BIST Macrocell Architecture

The total number of cycles necessary to perform a full test of a memory of 2^N words of mbits is $17N \times (\log_2 m + 1)$. The SRAM BIST macrocell implemented is represented in figure 3. The periphery of the BIST macrocells consists only of the buses between the Test Register and the rest of the chip. The Test Register is used only during test.

The launch of the test of the SRAM array and the control of the result is made through the BIST-BUS, activated by the BISTI block. Diagnostic capabilities have been added:

- In case of a failure detected in the SRAM array, the Test Register captures and holds the address location where this failure occurs, together with the incorrect data read from the SRAM.
- A Shift mode allows access to the SRAM buses through the scan path, to perform Read/Write of any word at whatever location.

Rather than making a block of fixed size, a generator has been built to adapt the test blocks to the size of the SRAM : AL (address width) and DA (data width) are the two parameters requested by this generator.

BIST BUS Description

The SRAM BIST Macrocell is connected to an internal BIST-BUS composed of 7 wires. This BIST-BUS is the link between the BIST Macrocells and the BIST Interface block; all the test actions are activated and controlled through this bus.

MTCK. Test clock. All the actions of the macrocell are synchronized by the MTCK clock. In a JTAG chip this signal is synchronous with the external Boundary-Scan Test Clock (TCK) from the Test Access Port. During the Test of the SRAM, the system clock (SCLK) and the Test clock MTCK have to be synchronous.

MTMS1/MTMS2. Test mode selection. 4 modes are encoded by these two wires: External Test, Self-Test, Shift-Mode and Normal Operations (see below).

MTRSTB. Reset (asynchronous) of the macrocell controller.

MTDI. Serial access input to internal test registers.

MTDO. Serial access output of internal test registers.

MEND. Signal indicating the end of the test.

Other control pin, not in the BIST-BUS:

FAIL. Indicates on line that a failure has been detected in the SRAM array.

Block description

FSM. Finite State Machine. Interfaces with the BIST-BUS and sequences all the actions according to the Marinescu algorithm and to the test mode selected by MTMS1/MTMS2.

REGADR. Up-Down Counter. Generates the addresses for the memory array.

DATASR. Signature Analyzer. Compresses the data read from the memory by polynomial division. A minimum length (at least 10 bits) of the analyzer is requested, if the length of the word stored in the SRAM is less than 10, the remaining inputs of the register are tied to a fixed potential. A primitive polynomial is chosen according to the length of the Signature Analyzer.

REGDAT. Data Background generator. It generates the data written in the memory.

REGCMD. Read/Write generator. It generates the Read/Write signal for the SRAM.

COMPAR. Comparator. Compares on-line the data read with the expected data.

All the registers included in these blocks have scan capability; they are used to build the Test Register. The registers in REGADR, DATASR, REGDAT and REGCMD are bypassed when the macrocell is in "normal" (= non test) mode.

Functional modes

These modes are entered after the first rising edge of MTCK based on the value of MTMS1 and MTMS2, except for Reset which is asynchronous.

1) Reset of the test register.

This is done by applying an asynchronous "0" on the signal MTRSTB. The test register is initialized, the macrocell is forced into its normal operating mode and does not interfere with the system functionality of the circuit.

2) Normal operation.

This state is either entered by applying a low pulse on MTRSTB or by applying the cod "01" on MTMS2/MTMS1.

3) Shift mode.

The MTMS2/MTMS1 wires are set to the shift mode ("10"). At each clock cycle the contents of the test register is shifted towards MTDO. At the same time, it can be loaded with a pattern to test the external logic of the macrocell (interconnections with the rest of the chip), or to test the SRAM array itself. When a pattern is shifted in the scan path, it is applied at the same time to its parallel outputs, excepted for the read/write command of the memory which is held in a safe state (read operation).

4) Self-Test.

The MTMS2/MTMS1 wires are set to the self-test mode ("11"); the macrocell takes the control of the SRAM buses. A seed is generated at the initialization phase of the self-test in the signature analyzer DATASR, and the test proceeds at each clock pulse until the end of test (see formula). Every data read during this phase is compressed in a Signature Analyzer (DATASR). When the sequencer detects that the test is completed, it sends an MEND signal and freezes the contents of the Test Register, even if the test clock continues to run.

A data comparator (DATA COMPAR) is also activated, it receives the data read from the memory and compares it on-line to the expected data generated by the DATA Write generator (REGDAT). The result of this comparison is sent to the sequencer, which sets the FAIL signal high and stops the test process if a discrepancy is found. Then the Test Register holds together the address where the mismatch occurred, the expected data and the actual data read out.

5) External test mode.

The MTMS2/MTMS1 wires are set to the external test mode ("00"). This mode is dedicated in the BIST-BST method to test the periphery of the macrocell, which is not activated by its self-test. In our case, this is the interconnections between the SRAM and the rest of the chip. The registers connected to the input buses of the macrocell (address bus, data input bus, R/W command) capture these buses at each rising edge of MTCK.

The register connected to the data output bus has different behaviour, depending on the internal state of the macrocell controller; this has been implemented for diagnostic purposes. If no self-test has been run since the last asynchronous reset, or if no fail has been detected on-line during the last self-test, this register is loaded with a fixed value ("0..0"), according to the BIST-BST method; otherwise this register captures the data present on the read-bus of the RAM array: this allows access to the RAM array, through the scan-path, to diagnose a detected fault.

Test results

During the test process, two cases can occur : either a fail has been detected on line by the data comparator and the FAIL flag is raised, or nothing has been detected, the MEND flag has been raised and the FAIL flag is lowered. When the self-test is completed upon one of these conditions, the test register retains its state, even if the test clock is always running. Afterwards, the contents of this register can be shifted out to verify the signature: for a good device it depends only on the number of addresses and on the data width, and can be theoretically calculated or simulated. Following the BIST and BST method, this signature is shifted serially into the Centralized Signature Analyzer.

If no FAIL has been detected (FAIL flag = 0), it does not mean the SRAM is fault-free; if any bit of the signature held in the Test Register is not as expected, the self-test is unsuccessful.

If a FAIL has been detected (FAIL flag = 1), the test is unsuccessful; the Test Register holds diagnostic data: REGADR holds the address where the fail occurred, DATASR holds the faulty data read in the memory at this location and REGDAT holds the data expected.

The signature resulting from a successful self-test has some characteristics that cannot be found in this diagnostic data.

Faults covered in RAM BIST Macrocell

The 17N Marinescu algorithm is calculated to detect several functional failures in the RAM circuitry (6), (80), including the matrix and all the glue logic:

1) Defects of the matrix:

→Stuck-at of one or several cells.

→Coupling between cells: a transition 1→0 or 0→1 on each cell i changes the state of one or several cells j , depending of the state of the cell j .

2) Defects of the decoding logic:

→The decoder does not address the right cell, but eventually other cells.

→The decoder addresses several cells, including the right one.

3) Defects of the Read/Write amplifier:

→Stuck-at of the Read/Write command.

→Stuck-at, open, shorts of the Data buses.

Concerning the RAM BIST Macrocell as a whole, the faults in the BIST circuitry have to be taken into account. A comparator checks, at each read action, the data coming from the memory against the expected data. To use only this feature may hide some problems when dealing with multiple faults, some defects in the comparator or in the data generator can be masked, or they may hide defects in the RAM array. On the other hand, by using only signature analysis, the risk is fault masking. The combination of both technics, at a low expense, enhances the fault coverage of the whole macrocell.

Logic Synthesis Design - Results

The test block is designed as a generator in VHDL language with the number of address wires (AL) and the number of data bits (DA) as parameters. This high level description can be efficiently simulated to calculate the good signature. This program

Table II. SRAM BIST Macrocell features

Address Size	Data	Area Width	Clock mm ²	Frequency Mhz (worst case)
4k	32	4.66	34	
8k	16	3.07	50	
1k		8	2.07	50

can be synthesized stand-alone or be included within the complete description of a chip in a full synthesis design approach. The generated netlist can be mapped onto Standard-Cell or implemented with auto-layout tools.

Several examples have been implemented. Table II summarizes their main features for a CMOS technology 1.2μm, 2 metal. The additional cost in area is

less than 10% of the RAM array itself for these cases, and the additional transit time is less than 1ns, worst case, on each bus.

4. SRAM TESTER CHIP

A chip dedicated to test static RAM board arrays is proposed. Experimental chips have been made for this purpose (9), (10), here is shown a real versatile catalogue product, compatible with the Hierarchical Self-Test concept and supporting the RUN-BIST instruction with the BIST-BST method.

General Description

The purpose of this chip is to help periodic maintenance of a Static RAM Board by testing it without external equipment, in off-line mode. The test consists in performing Read/Write in the SRAM memory array with a predetermined algorithm (Marinescu 17N). This chip conforms to the IEEE 1149.1 standard and supports the RUNBIST instruction. All the operations of the chip are made through the JTAG bus via special instructions, only one clock, the JTAG clock is required.

In case of failures detected in the SRAM array, the mapping of the faulty chips is provided, as well as the address where the last failure occurs with the faulty data read out.

After lowering the TRSTB pin or having placed the internal JTAG TAP-INTERFERENCE in the TEST-LOGIC-RESET state, all the outputs connected to the RAM buses are put in high impedance, in order not to disturb the system function of the board. This can be ensured if the concerned chips support the HI-Z instruction, as proposed in supplement to JTAG standard.

49 general purpose pins can be used as Address, Data and Chip-Select buses to interface with the RAM array. Their partition is determined by a pattern loaded in a dedicated Test Data Register (CONFIREG), activated by a JTAG instruction (CONFIG). See figure 4 in the SRAM TESTER CHIP Block Diagram.

Functional Description

The SRAM tester Chip performs test of Static RAM array according also to the MARINESCU 17N algorithm. Thus, the faults covered by the algorithm in the SRAM board array are the same than for embedded static RAM. The SRAM tester chip generates the Addresses, Chip-Selects and Data according to this algorithm, considering the capacity and organization (number of chip selects) of the RAM array. The access to the RAM array is made in parallel with the functional RAM buses, which means that all the drivers connected to these buses should be put in high impedance state during RAM self-test, but does not introduce an additional delay during system access. Each step of the algorithm needs 4 clock cycles to be performed.

This algorithm is run the number of times corresponding to the width of the Chip-Select Bus, and for each time with different Data-Backgrounds, according to the width of the Data words. The required total number of cycles to perform a full test of a memory of 2^N words of m bits is $68N \times (\log_2 m + 1)$.

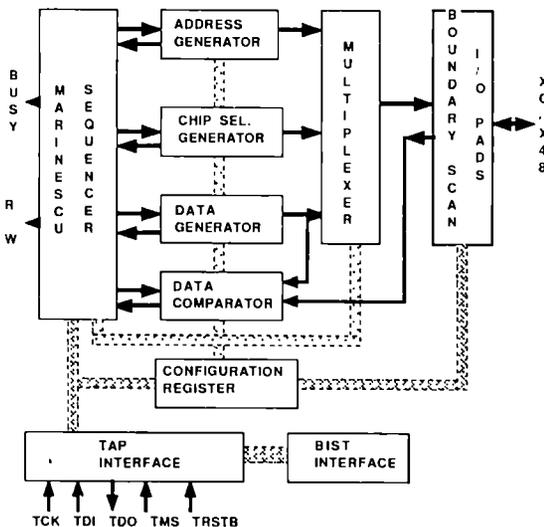


Figure 4. SRAM TESTER Chip Block Diagram.

Activation of SRAM Tester Chip

All the functions performed by the chip are controlled via the 1149.1/JTAG bus, according to the instruction loaded in the Instruction Register; the JTAG clock TCK is the only necessary signal during the test-phase. When not used (The TAP-Controller being placed in the Test-Logic-Reset state by activating TMS and TCK or by lowering TRSTB), all the Bidirectional Pins and Three-States Outputs are in High Impedance.

Pin Usage

The Partition of X0-X48 pins as Address bus, Data bus or Chip Select bus or Inactive is determined by a pattern loaded into a test data register (CONFIREG), through a JTAG instruction (CONFIG) giving access to it.

In binary MSB → LSB (d_0 entered first in serial) the data to preload is:

$a_5a_4a_3a_2a_1a_0c_5c_4c_3c_2c_1c_0d_5d_4d_3d_2d_1d_0$, with $a_i, c_i, d_i = '0'$ or $'1'$ and the number of Address wires (NA), Chip-Select wire (NC) and Data wires (ND) being supposed to be respectively

$$NA = \sum_0^5 a_i 2^i, \quad NC = \sum_0^5 c_i 2^i, \quad ND = \sum_0^5 d_i 2^i$$

The only constraints on NA, ND and NC are:

$$\begin{aligned} 4 \leq NA+ND+NC \leq 49 & \quad ; 2 \leq NA \leq 40 \quad ; \\ 1 \leq ND \leq 46 & \quad ; 1 \leq NC \leq 46 \end{aligned}$$

The ND Data pins are located on the chip pin-out from X_0 to X_{nd-1} , the NA Address pins are from X_{nd} to $X_{nd+NA-1}$, the NC Chip Select pins are from X_{48} to X_{48-NC} . The other pins X_i , if exist are in high impedance.

Static RAM test launch

After having programmed the X0-X48 pins, the test of SRAM array can proceed. For this, it is necessary to load the appropriate instruction (TESTRAM) into the Instruction Register by using the JTAG protocol. When the TESTRAM instruction is active in the chip, and when the internal TAP is in the RUNTEST/IDLE state, the BUSY pin is lowered to indicate that the chip is taking the control of the buses. Then, at each four TCK clock cycles, a step of the algorithm is run; the first RAM chip is activated by lowering the first Chip-Select wire and the whole algorithm is run several times, each time with a different data background. Afterwards, the next chip-select is activated and the second RAM is tested, and so on until the whole array is tested.

Static RAM test Result

The Data read out from the RAM chip is checked inside the SRAM tester chip against the expected one. After completion of test, two signatures are located in the SIGNAREG and DIAGREG Test Data registers. If the TCK clock goes on running when the test is over, the results remain unchanged in the internal Test Registers.

SIGNAREG is selected by the TESTRAM instruction. If NC chips were tested, the first NC bits represent the mapping of faulty chips, if any; a good chip is characterized by a "1", a faulty one by "0".

DIAGREG is accessed through the DISGBUS instruction. It contains the state of the address bus when the last fault has been detected, together with a word indicating the location of the faulty bits of the data word read into the memory.

Test Architecture

The Static RAM tester chip is able to perform a RUNBIST instruction. This instruction generates a complete self-test of the chip. This self-test is entered when the RUNBIST instruction is active and when the TAP-CONTROLLER is in the RUNTEST/IDLE state; all that remains is to activate the TCK clock.

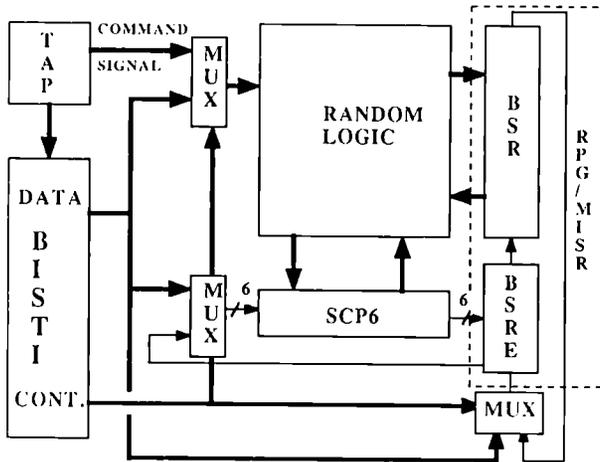


Figure 5. SRAM TESTER self-test structure.

The test architecture is represented on the figure 5, the main blocks are:

- **The TAP INTERFACE (TAP).**
- **The Boundary-Scan REGISTER (BSR, 101 bits).**
- **The BIST INTERFACE (BISTI).** A sequencer for the BIST function.
- **SCAN-PATHS (SCP6).** Set of functional registers capable of being configured as 6 parallel scan-paths, from 18 to 59 bits.
- **RANDOM LOGIC :** Constituted by the combinatorial part of the circuit. This block functionally inter-

faces with the TAP, the BOUNDARY-SCAN and the SCAN-PATHS.

- BSR Extension: (BSRE, 6 bits).

BSR and BSRE are used together as a single 107 bit Centralized Random Pattern Generator and Multiple Input Signature Register 9 (RPG/MISR), with $(x + x^2 + x^3 + x^5 + x^7 + x^{107})$ as primitive polynomial. Simulations have been made to check that no limit cycling (11) occurs.

The self-test flow consists of three phases, automatically sequenced by the BISTI block:

- 1) Serial entry of a seed into the Random Pattern Generator/Multiple Input Signature Analyzer.
- 2) Serial entry into the Scan-Paths of deterministic patterns to test the Set/Reset function.
- 3) Configuration of the circuit into different modes, by forcing the command signals normally issued by the TAP, and random testing of the internal logic during these modes.

In this phase two steps occur:

- 1. Serial entry of random values into the 6 scan-paths (SCP6): 6 different sources of random bits are obtained from 6 parallel outputs of RPG/MISR. At the same time, the contents of the scan-paths are applied to parallel inputs of the RPG/MISR for signature analysis.
- 2. Application of the parallel output of SCP6 and of RPG/MISR to the random logic. Then, after stabilization, capture of the random-logic outputs in the SCP6 register and parallel signature computation at the inputs of RPG/MISR.

After that, step 1 proceeds again, until a sufficient number of patterns has been entered; this number depends of the mode forced on the command signal, in order to optimize the test length without compromising its efficiency.

When all these steps are completed, the final signature is in the Boundary-Scan Register, and can be shifted-out by entering the SHIFT-DR state in the TAP controller.

RUNBIST in Practice

16×10^6 clock cycles are necessary to achieve the complete sequence, which corresponds to more than 260,000 parallel vectors applied to the logic. Several fault simulation runs have been launched up to 100,000 clock cycles in the RUNBIST mode, which has given a fault-coverage rising up to 84.3% for stuck-at faults. The fault-coverage versus clock cycles number curve obtained during these different fault simulation being not flat between 50,000 and 1000,000 clock cycles, we can expect an even better fault coverage 16M cycles.

Another instruction: RRUNBIS (for Reduced Run-Bist) enters the same process, but for only 1600 clock cycles; the fault coverage is not satisfactory with this instruction (55%), but this instruction has been implemented to be able to simulate the complete self-test process during the design phase, because to simulate 16 Mega cycles is not possible! That means also that the final good signature is known only when the silicon is available.

Design for Testability at VHDL level

The core of chip has been described in VHDL in order to be synthesizable. Design for testability modifications have been made at this high level description, without knowledge of the future mapping in gates. This method implies following precise design rules, in order to avoid clock skew or unpredictable spikes on sensitive signals (clock, reset wires), which can affect the contents of registers.

An analysis has been made also to evaluate the controllability and observability of the functional links (control and data signals) between blocks. Modifications have then been made, and controlled by VHDL simulation, to the finite state machine of the centralize sequencer to have an even distribution of these figures.

The modification consists of an appropriate coding of all possible states of the state diagram, knowing that these states are randomly accessed during the RUNBIST sequence.

Design Methodology - Results

The chip can be divided in three parts:

- **The Pad ring** : It is made with High Power Pad Buffers.
 - **The Boundary Scan Ring** : The Boundary-Scan Registers are fully customized and can be abutted to the PADS, in order to build a complete ring.
 - **The Core logic** : This part has been entirely made by synthesis, starting from a VHDL description, and mapped onto a $1.2\mu\text{m}$ CMOS standard cell technology. The TAP INTERFACE has been produced by a generator written in VHDL.
- First silicon samples in BICMOS $1.2\mu\text{m}$, 2 metal, are available since September 1991.

Design data.

- Transistor Count	: 54,000	
- Overall chip area:	: 55 mm ²	
- Boundary Scan Register	: 5.9 mm ²	→ 10.7%
- TAP INTERFACE	: 0.6 mm ²	→ 1.1%
- BIST controller	: 0.9 mm ²	→ 1.6%
- Scan Path Overhead	: 0.4 mm ²	→ 0.7%
- BIST-BST overhead	: 7.8 mm²	→ 14.2%

The overhead due to the Boundary-Scan Register is large, because each I/O (for a total of 50) has an independent Tri State Control cell, to fully comply with the IEEE standard. The abutting Full-Custom cell approach has reduced this area by a factor of more than 2, compared to a Standard Cell based design.

5. Conclusion

In this paper, we have presented practical implementations following the Hierarchical Self-Test Concepts. Tangible results are obtained: a BIST macrocell to test embedded static RAM has been investigated and added to our library; a versatile, JTAG compatible RAM board Tester Chip will be available as product in the near future. This work shows also the benefits of such a technic:

- The IC Designers increase their design productivity by using a straightforward method indicating how to implement a BIST scheme and how to activate and control the different BIST resources. This allows to capitalize on experience from different design departments of the same company, by having a common approach for the BIST library elements.
- The System Designers may have a real multi-levels approach, where the test of each unit is taken in charge by documented specific test elements, down to the silicon level.
- Everybody can concentrate on the functional architecture of his subject, test and diagnostic features can be automatically added and linked.

Future work

Next steps will be to define the applicability area of this technic at the IC level and then to derive other self-testing macrocells to increase our BIST macrocell library.

Other JTAG service IC's like a Test Processor to take charge of the self-test of a board and make the interface between the MTM bus and JTAG bus will be also investigated.

Acknowledgement

This work has been partially supported by the European Strategic Programme for Research and Development in Information Technology (ESPRIT II) under contract #2478, "Research into Boundary-Scan Test Implementation".

Mr Parot from THOMSON-CSF/SCFT has proposed judicious suggestions for the SRAM TESTER Chip.

REFERENCES

- (1) J. Maierhofer, "Hierarchical Self-Test Concept based on JTAG standard". *Proc. Int'l Test Conf.*, 1990, pp.127-134
- (2) R.P. van Riessen, "Design and Implementation of a Hierarchical Testable Architecture Using The Boundary Scan Standard". *Proc. Europ. Test Conf.*, 1989, pp.112-118
- (3) F. Beenker et al., "Self-Test for SRAMs", *IEEE Design & Test of Computers*, Feb 1989 pp.28-34
- (4) M. Nicolaidis, "An efficient Built in self-test Scheme for functional Test of embedded RAM"., *15th FTCS Ann Arbor*, 1985
- (5) R. Dekker et al., "A Realistic Self-Test Machine for Static Random Access Memories". *Proc Int'l Test Conf.*, 1988, pp.353-361
- (6) M. Marinescu, "Simple and Efficient Algorithms for Functional RAM Testing". *Proc. Int'l Test Conf.*, 1982, pp. 236-239
- (7) R. Dekker, "Fault modeling and Test Algorithms Development for Static Random Access Memories". *Proc Int'l Test Conference*, 1988, pp.343-352
- (8) R. Nair et al., "Efficient Algorithms for testing semiconductor random Access Memories". *IEEE Transaction on Computers C27*, 1988, pp.343-352
- (9) J.P. Mucha et al, "Self-Test in a standard cell Environment". *IEEE Design & Test*, pp.35-41.
- (10) W. Daehn et al., "A Test Generator IC for Testing Large CMOS-RAMS", *Proc. Int'l Test Conf.*, 1986, pp.18-24
- (11) M. Ohletz et al, "Overhead in Scan and Self-Testing Designs". *Proc. Int'l Test Conf.*, 1987, pp.460-470

PLANET MOVPE EQUIPMENT: A BREAKTHROUGH IN INDUSTRIAL PRODUCTION OF SOPHISTICATED EPITAXIAL LAYERS

L. Hollan and P.M. Frijlink
LABORATOIRES d'ELECTRONIQUE
PHILIPS
22, avenue Descartes
F-94453 Limeil-Brevannes Cedex
Tel: (33).1.45699610
Fax: (33).1.45694088

J.M. Marchal and Ch. Waucquez
POLYFLOW
16, place de l'Université
B-1348 Louvain la Neuve
Tel: (32).10.452861
Fax: (32).10.453009

H. Juergensen and G. Strauch
AIXTRON
Kackertstrasse 15-17
D-5100 Aachen
Tel: (49).241.89090
Fax: (49).241.890940

J.F. Hernandez-Gil Gomez
TELEFONICA I&D
Emilio Vargas nº 6
E-28043 Madrid
Tel: (34).1.3374115
Fax: (34).1.3374712

G.A. Acket and H.P.M. Ambrosius
PHILIPS OPTOELECTRONICS CENTRE
NL-5600 MD Eindhoven
Tel: (31).40.742993
Fax: (31).40.743859

E. Calleja Pardo
ETSI TELECOMUNICACION
Universidad Politecnica Madrid, Ciudad
Universitaria,
E-28040 Madrid
Tel: (34).1.5495700 Fax (34).1.5439652

ABSTRACT

The aim of Esprit Project Planet is to develop a high throughput, industrial MOVPE (Metal Organic Vapour Phase Epitaxy) equipment suitable for the fabrication of the complex III-V heterostructures required in micro- and optoelectronics.

The work is carried out in collaboration between AIXTRON, the world leading MOVPE equipment manufacturer, PHILIPS and TELEFONICA involved in MOVPE growth, POLYFLOW and the TECHNICAL UNIVERSITY of MADRID.

The PLANET equipment is based on a completely new concept of an MOVPE system allowing simultaneous growth of 7 wafers in planetary motion, using gas foil rotation. An industrial system has been defined and constructed, and improvements are being implemented. A mathematical model is also developed to simulate the growth.

The performances of the PLANET equipment are outstanding: the layer thickness uniformity on 2" and 3" wafers is within +1% and the composition is well reproducible within 0.5%, whilst the average surface defect density is $5/\text{cm}^2$.

The evaluation of the PLANET system in microelectronics is made with HEMT's; high values of cut-off frequencies (65 GHz) are obtained with low dispersion. In optoelectronics, the very low laser current densities ($400 \text{ A}/\text{cm}^2$) obtained also show the excellent quality of the as grown material.

The PLANET type equipment is already commercially available from AIXTRON. Based on PLANET MOVPE material, a (PM)HEMT foundry process will be made commercially available at Philips Microwave Limeil, whilst Philips Optoelectronics Centre intends to industrialise mass production of lasers and Polyflow sells a software package for growth simulation.

INTRODUCTION

III-V devices and IC's are required for an increasing number of applications that are progressively extending from professional systems to consumer goods. On the other hand, the totality of the opto-electronic and an increasing part of the microelectronic applications require epitaxial materials. They are summarized in Table I.

FIELD	MARKET	CHALLENGE
	Direct Satellite Broadcast	Low Noise HEMT's
Microelectronics Microwave	Professional : Super computer Military, Telecom., Space Cellular Telephone Automotive Optical Disk Components	High Frequency low noise ASIC's High Power Efficiency High Frequency Radars Low Cost Lasers
Optoelectronics	Fiber Optics Front Ends Laser Printers Barcodes Readers High Density Optical Storage	Low Cost Lasers and Detectors High Power Lasers Low Cost Visible Lasers Short Wavelength Lasers
Integrated Optoelectronics	I. O. Functions Switches	Large Area Quaternary Structures
Miscellaneous	Solar Cells Night Vision Tubes	Large Area Efficiency Heterostructures Large Area Defect Free Heterostructures

Table I: Applications based on III-V epitaxial materials

The common feature is that all those applications require specific, very well controlled heterostructures of various materials, with high purity, sharp transitions, the best uniformity and reproducibility. For consumer, or even for some professional applications, high throughput is also demanded. In the present phase of building and positioning this new, strategic industry, the availability of the relevant epitaxial growth equipment is of prime importance.

The aim of Esprit Project PLANET is to develop and make commercially available a high throughput, multiwafer industrial MOVPE (Metal Organic Vapour Phase Epitaxy) equipment and to demonstrate the capability of such a system with state of the art devices in the most exacting application areas: high electron mobility transistors (HEMT) in microelectronics, short wavelength (AlGaAs and InGaAlP (red)) lasers, and guided optics in optoelectronics.

COOPERATION

The multiple scientific, technological and industrial aspects are addressed by a consortium, by sharing the work between industrial partners (AIXTRON, the world leading MOVPE equipment manufacturer), companies involved in MOVPE growth and the use of material (PHILIPS: LEP and Philips Optoelectronics Center (POC), TELE-

FONICA) experts in numerical simulation (POLYFLOW) or in III-V material characterization (TECHNICAL UNIVERSITY OF MADRID). The cooperation scheme is depicted on fig. 1: it shows the contribution of the partners and indicated the main feed-back loops aiming at the improvement of the prototype system both in performance and in user friendliness.

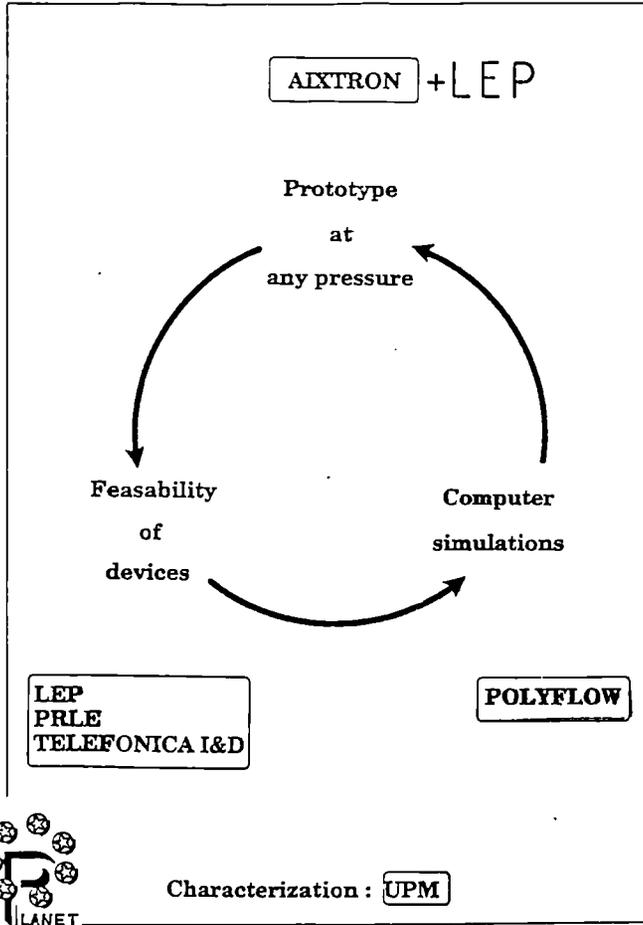


Fig. 1: The interactive cooperation scheme of the Planet program

THE EQUIPMENT CONCEPT

The equipment is based on a completely new concept of an MOVPE system.

Mainly two multiwafer geometries have been applied: the barrel reactor geometry (1), (2), and the rotating disc reactor (3). In 1988, the new concept of MOVPE reactor of this ESPRIT project was introduced (4), for simultaneous growth of very uniform epitaxial layers on 7 wafers in planetary motion, using a circular growth chamber with horizontal geometry and radial gas flow (fig. 2). In this concept, the radial flow implies a decrease of the gas velocity along the flow direction, making it possible to realize an almost linear decrease of the growth rate with increasing radius, by depletion of the gas phase.

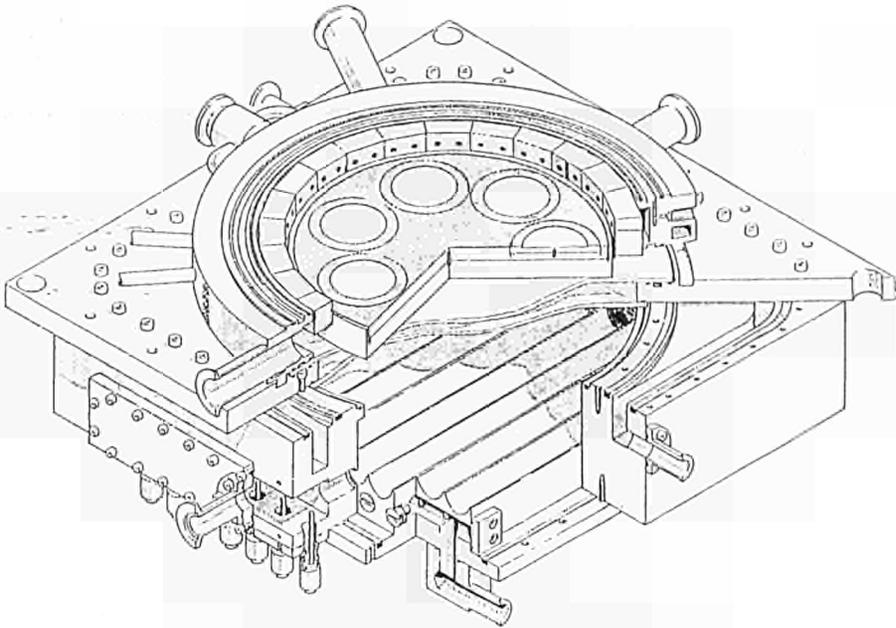


Fig. 2: Schematic view of the planetary reactor

Another important aspect of this new concept is the use of gas foil rotation to ensure the planetary rotation of the graphite substrate holder discs (fig. 3): centrally fed radial gas streams are used to provide levitation and frictionless rotation, and thus avoids mechanical particle generation.

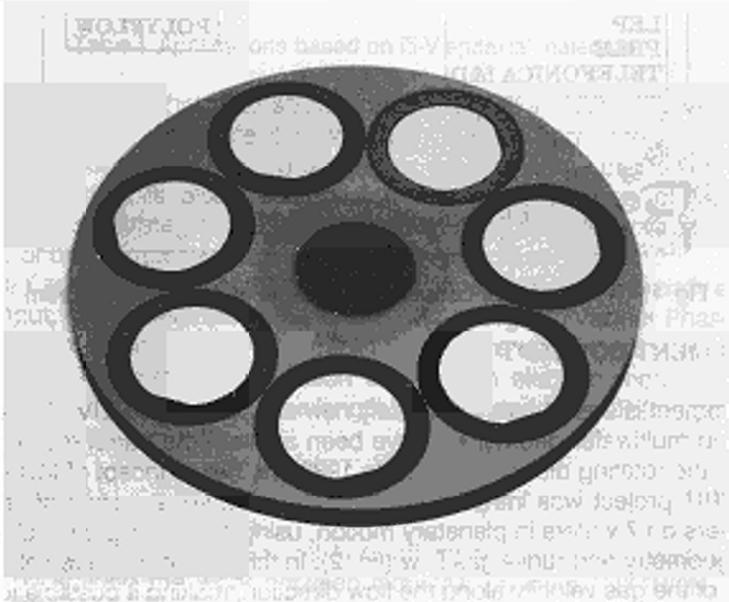


Fig. 3: Planetary motion of the wafers insured by gas (H_2) flow rotation

RESULTS AND ACHIEVEMENTS

1. MOVPE equipment development

Starting from the above described new reactor concept, a complete, user friendly, industrial reactor system has been defined, constructed and installed at POC during the first year following a tight time schedule. The detailed definition of the requirement and the final reactor design of the MOVPE system have been completed between AIXTRON, LEP and POC, taking the advantage of the flow simulations carried out by POLYFLOW. According to the program, improvements of the system have also been implemented. This has been achieved in good cooperation, taking into account the experience gained by the users and the equipment manufacturer. It concerns various parts: the infra-red heating system, the gas injection (example to substitute for bellows), the exhaust filter (disposal of a cyclon trap), the susceptor (control of rotation through a view port), etc... The improvement the easiest to illustrate here as an example concerns the sinking of the wafer into the substrate holder as it is illustrated on fig. 4. The strong enhancement of the uniformity near the wafer edge is quite spectacular, on the other hand, the comparison between the experimental results and the calculated ones shows both the validity of the models and the accuracy of the measurements.

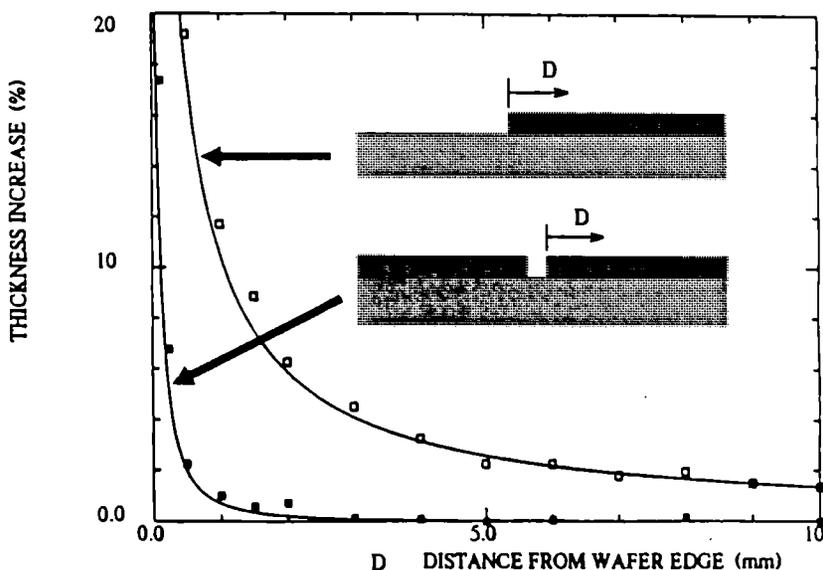


Fig. 4: Comparison of the calculated deviation of the arrival rate of group III element bearing species near the wafer edge normalized to the arrival rate for a flat surface (full lines), with the experimentally observed normalized growth rate (dots) for two cases with straight wafer edges as shown in the inserts.

The reactor allows the growth of GaAs, GaAlAs, GaInP and AlGaInP at any pressure. It is worth to point out here that a clear advantage of the MOVPE technology over MBE is its capability to grow in the same equipment phosphorous containing compounds. The system consists of three cabinets: the electronic control, the gas blending and the reactor glovebox: an overview of the prototype AIX 2000 planetary system, now commercially available is given in fig. 5.



Fig. 5: Front view of the Planet type AIX 2000 prototype MOVPE system.

The original and the new prototypes have proven to work very successfully and the results will be shown in the next paragraphs. Meanwhile, the developed AIX 2000 system has been sold to several industrial customers.

2. Numerical simulation

A good illustration of the advantageous conditions for cooperation opened by the Esprit context and fruitfully fulfilled by the partners is given by the interactive computer simulation support provided by POLYFLOW.

A mathematical model is developed to simulate the growth in the Planet reactor. The method is based on the latest Finite Element technology. Fig. 6 shows typical results that can be obtained from the 2D 1/2 model (three velocity components on a two-dimensional domain): velocity, temperature and concentration plots. From the concentration, the growth rate as a function of the position can be calculated. As it is shown elsewhere in this paper, these simulations have been used in the optimization of the reactor as well as that of the growth conditions.

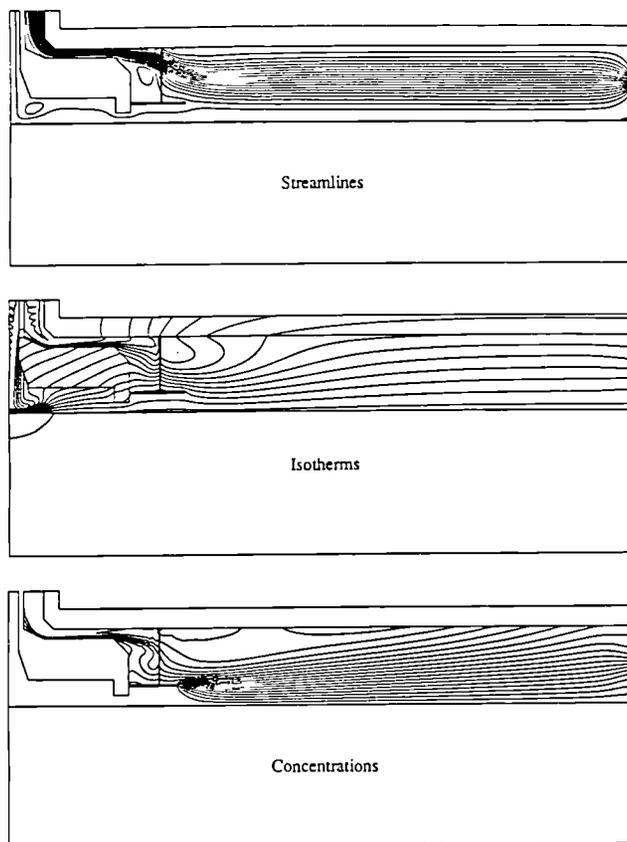


Fig. 6: Typical results of numerical simulations: streamlines, isotherms and concentrations.

3. Growth performances

Uniformity

The uniformity of the deposited layer thickness and of the doping levels have first been demonstrated in the original work (4) on 2" substrates at atmospheric pressure, to be within $\pm 1\%$ which remains the state of the art.

The uniformity obtained in the new system at a pressure of 200 and 100 mbar at various flow rates on 2" wafers has been described recently (5). In agreement with the simulations, the results confirm the original performances ($\pm 1\%$).

In parallel, the capability of the Planet system on 3" substrates has been demonstrated. The growth rate as a function of distance from the reactor center for different flow rates, calculated by numerical simulation is shown in fig. 7. The corresponding thickness uniformity on rotating 2" and 3" wafers depicted in fig. 8 a and b shows that, with the original conditions, the $\pm 1\%$ uniformity could not be achieved. The deflector ring at the entrance was therefore modified and the resulting thickness profile measured on a 3" wafer is shown in fig. 9; the thickness uniformity is accurate to $\pm 1\%$ up to 2 mm of the wafer edge, as well as the sheet resistance (dopant) uniformity shown on fig. 10.

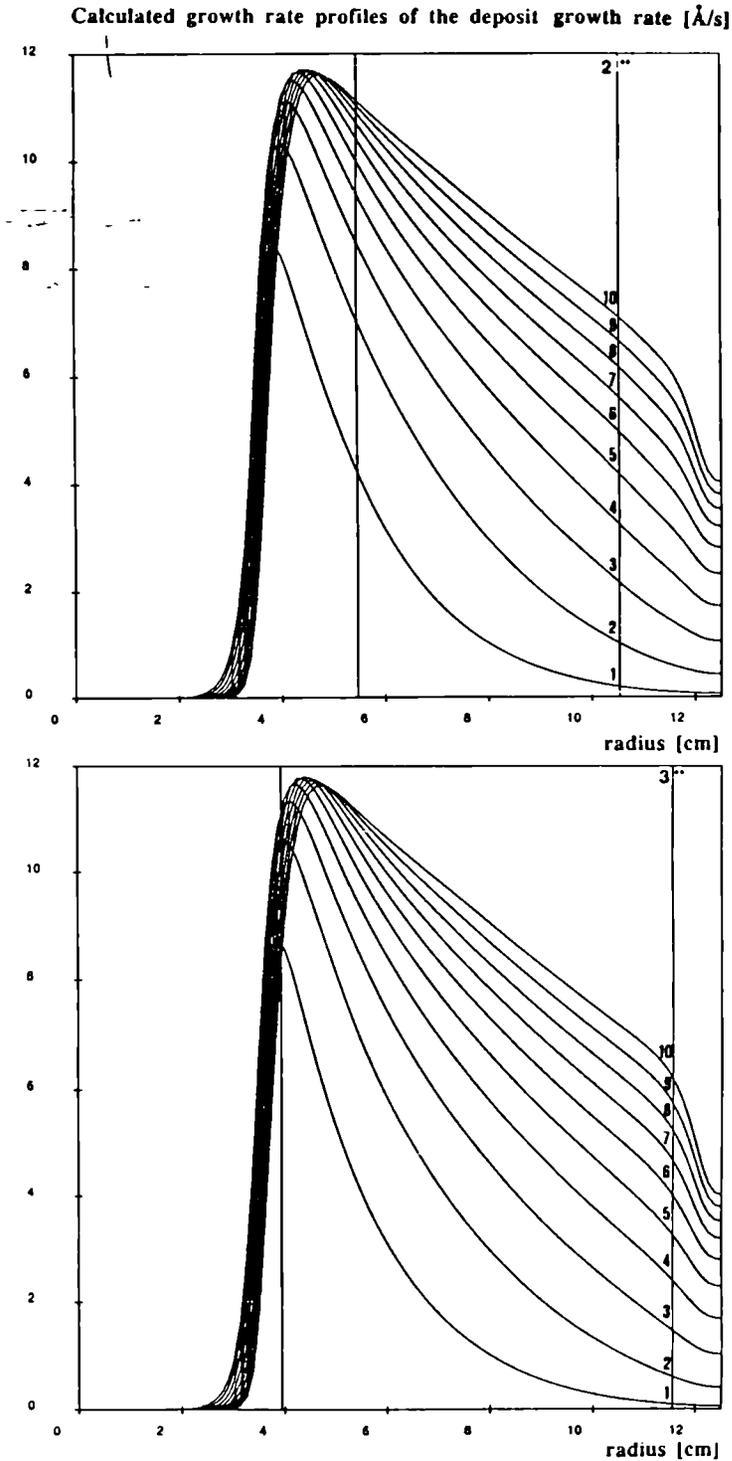
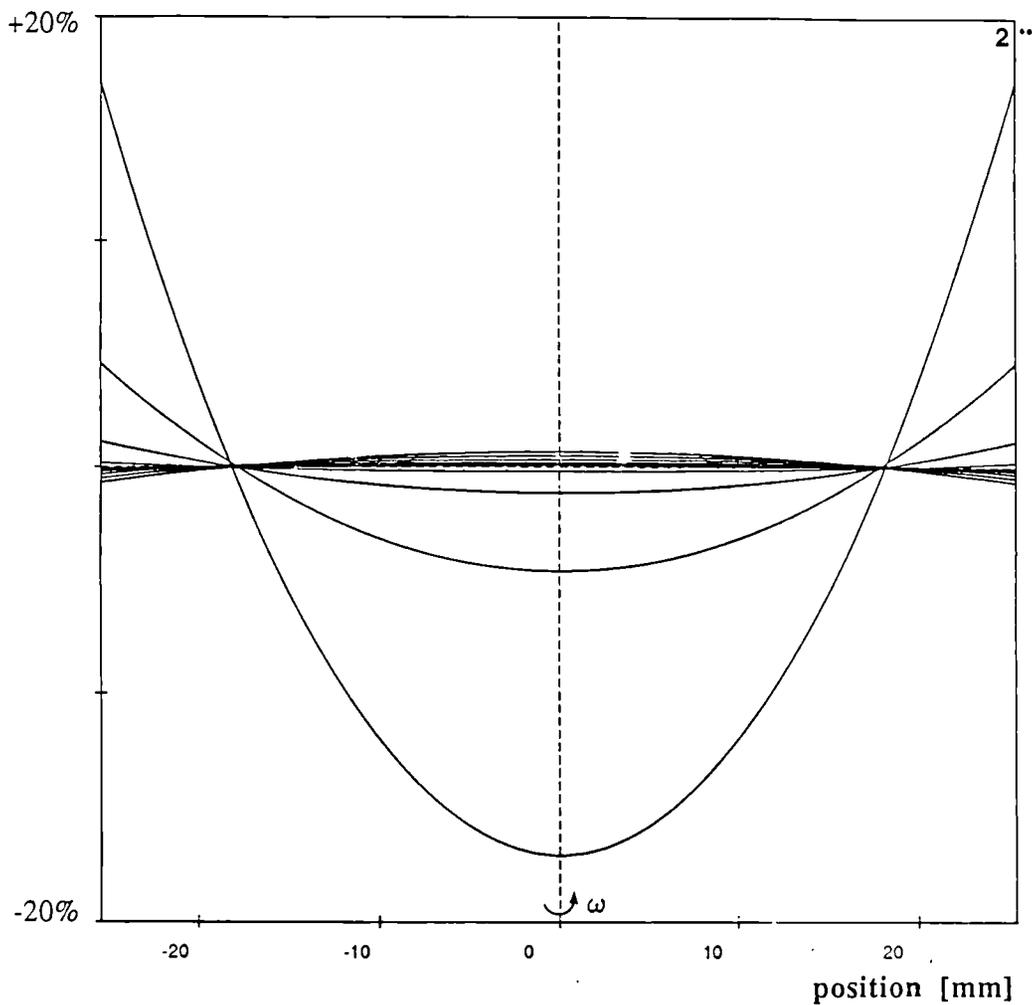


Fig. 7: Calculated growth rate as a function of the position in the reactor. The optimal positions for 2" and 3" wafers are indicated.

Calculated normalized thickness profile on rotating wafer

relative thickness [\pm %]

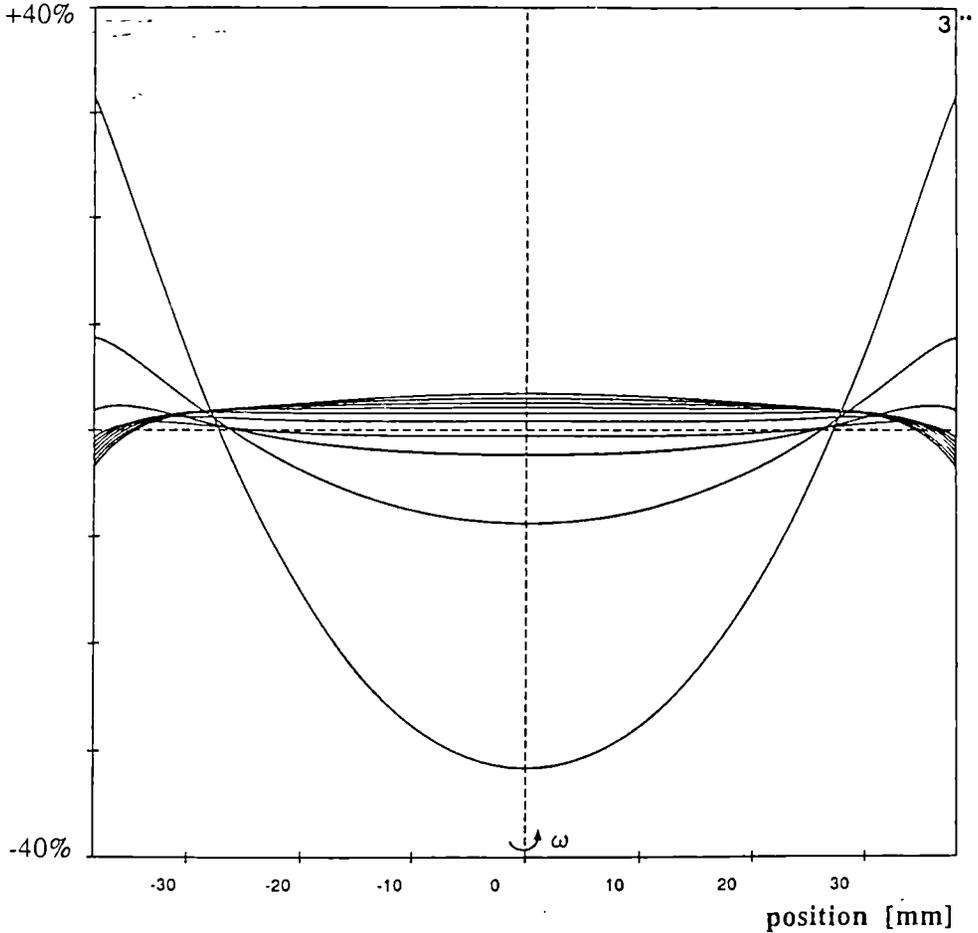


Total flow rates | curves | % standard deviation

.1 Qnom		1		+ - 17.1531
.2 Qnom		2		+ - 4.5550
.3 Qnom		3		+ - 1.1335
.4 Qnom		4		+ - 0.1867
.5 Qnom		5		+ - 0.0583
.6 Qnom		6		+ - 0.1294
.7 Qnom		7		+ - 0.2050
.8 Qnom		8		+ - 0.3308
.9 Qnom		9		+ - 0.4982
1.0 Qnom		10		+ - 0.6795

Calculated normalized thickness profile on rotating wafer

relative thickness [\pm %]



Total flow rates | curves | % standard deviation



.1 Qnom		1		+- 31.6358
.2 Qnom		2		+- 8.7782
.3 Qnom		3		+- 2.3500
.4 Qnom		4		+- 0.7574
.5 Qnom		5		+- 1.2539
.6 Qnom		6		+- 1.6818
.7 Qnom		7		+- 2.1340
.8 Qnom		8		+- 2.5945
.9 Qnom		9		+- 3.0529
1.0 Qnom		10		+- 3.5045

Fig. 8: Growth rate profiles on rotating wafers corresponding to fig. 6: a) 2 inch, b) 3 inch wafers.

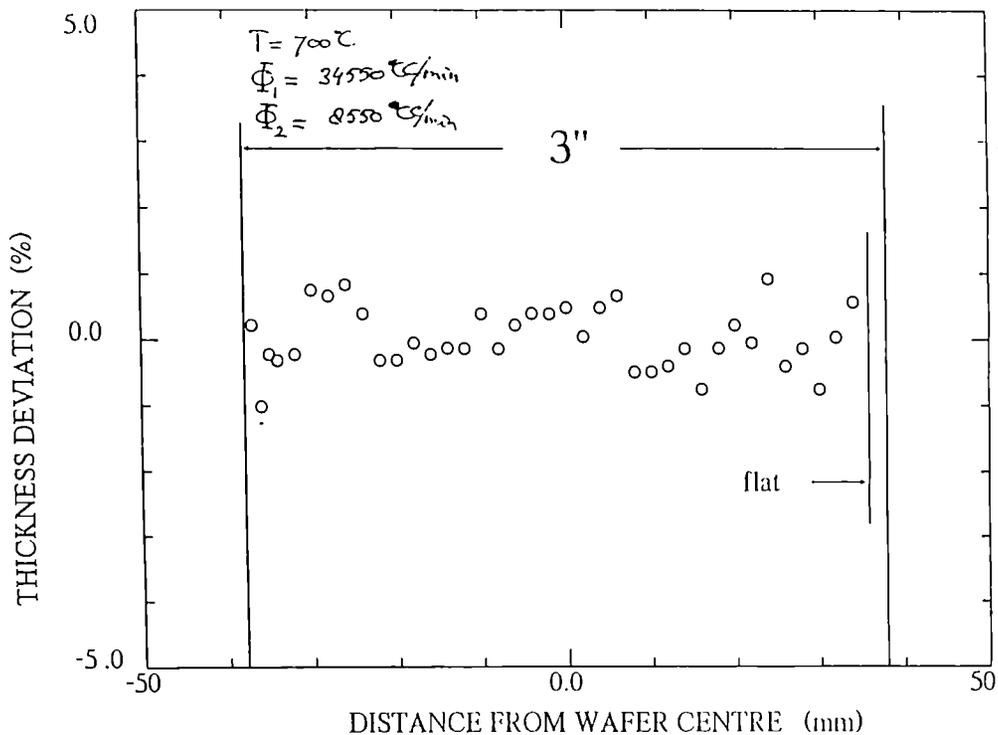


Fig. 9: Experimental thickness uniformity on 3 inch wafers.

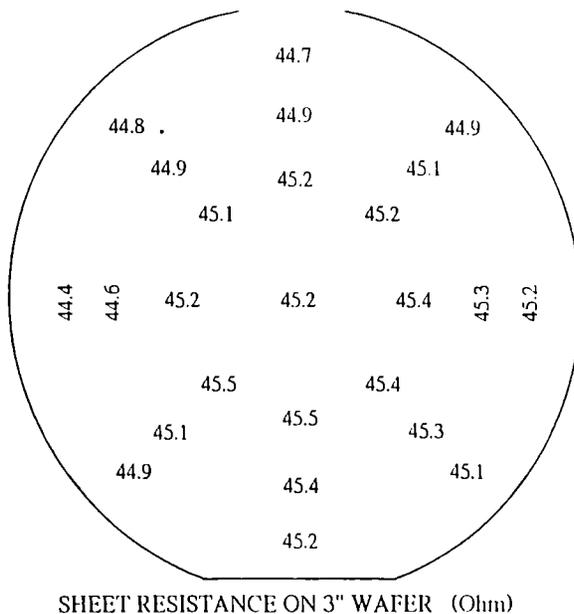


Fig. 10: Cartography of the sheet resistance of an epitaxial layer on 3 inch wafer.

Reproducibility

The layer uniformity is in fact inherent to the planetary rotation concept (when the growth rate variations are linear) and it proved to be reliable over more than 700 runs carried out in the original reactor.

A more stringent criteria is the reproducibility of the composition of ternary multi-layers, especially of the In concentration in GaInAs. As an example, the indium concentration of pseudomorphic HEMT (high electron mobility transistor) structures realized in 25 consecutive runs is depicted in fig. 11; it shows that even for this 100 Å thick layer the alloy composition is well reproducible within $\pm 0.5\%$.

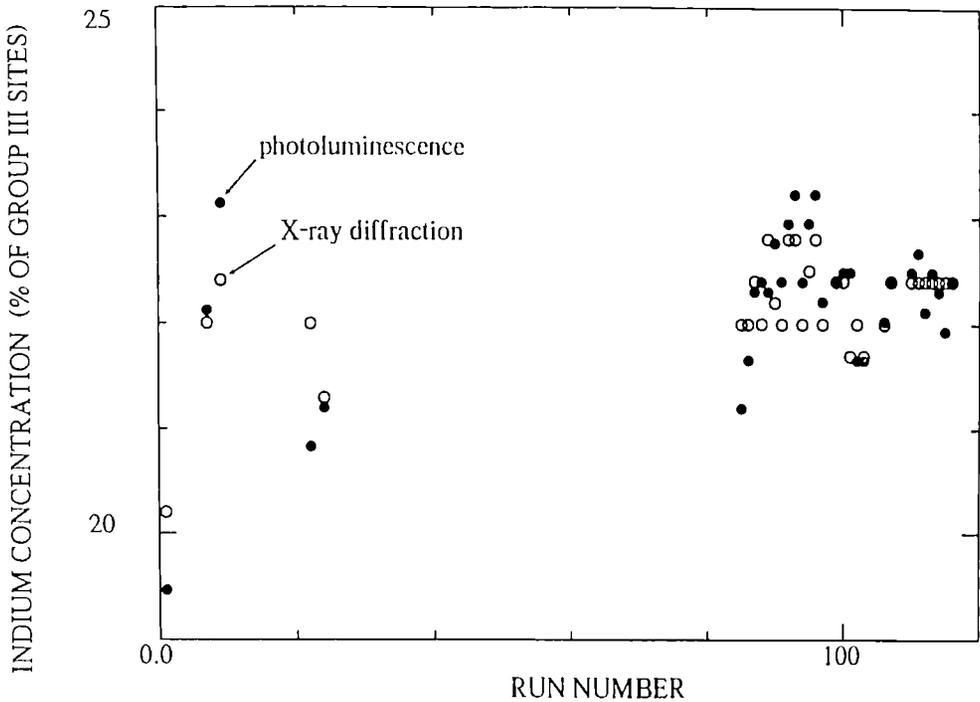


Fig. 11: Indium concentration as a function of run number in a 100 Å thick pseudomorphic GaInAs layer (see structure at fig. 12).

Defect density

For industrial applications it becomes more and more important to achieve the lowest possible surface defect density. The gas foil rotation provides an inherent advantage to the Planet MOVPE system to achieve this. By optimization of the substrate preparation and of the operational procedure, the following results have been obtained.

A record low value of 0.5 defects/cm² (10 over 2") has been observed by Tencor surfscan over the full range of defect sizes (0.18 μm to infinity). This low value does not yet characterize our average surface cleanliness, it indicates that the local MOVPE growth process in this reactor itself does not add more defects than above.

The actual average value has been established in another series of experiences: the histogram depicted in fig. 12 shows the defect density measured on wafers of 25 subsequent runs.

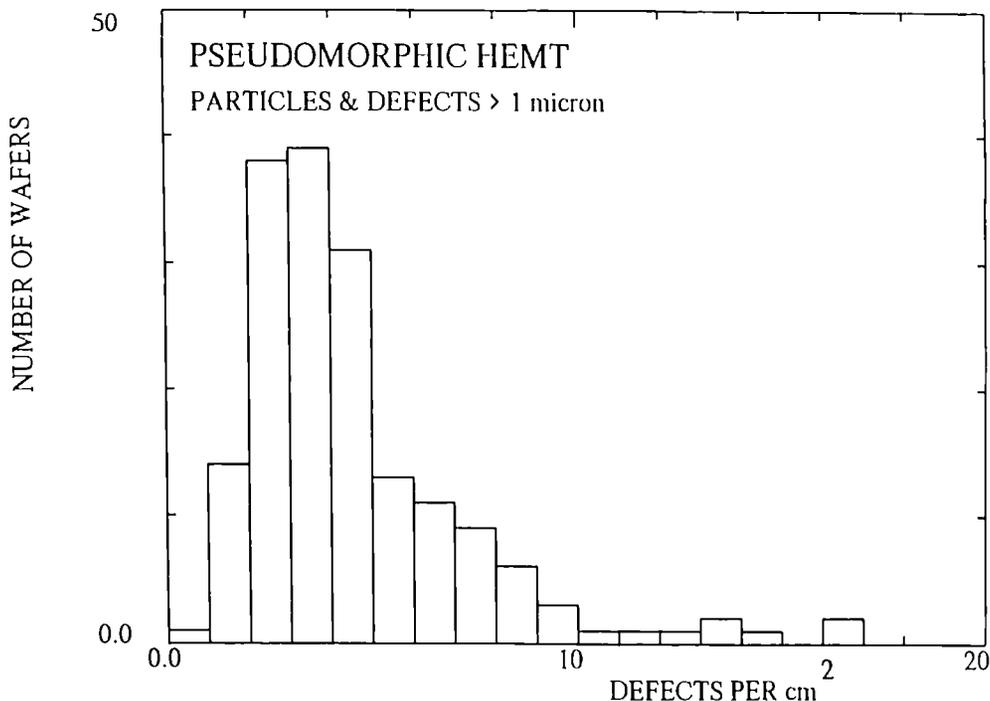


Fig. 12: Count of particles and defects on a typical wafer after epitaxy.

It should be pointed out that both the record and the average values represent the best results reported in MOVPE so far and on the other hand, are much better than the best achievements in MBE.

4. Evaluation of the reactor in microelectronics

The use of multiwafer reactor is mainly justified by the necessity to feed industrial device fabrication. The devices chosen to demonstrate the capability of the PLANET MOVPE system in the field of microelectronic applications are conventional and pseudomorphic HEMT's. The reason is that the two dimensional electron gas (2 DEG) structure requires a very tight control of the thicknesses and of the composition transitions, practically at the level of $\pm 10 \text{ \AA}$ only (see fig. 13). Furthermore, the pseudomorphic HEMT requires the mastering of a very thin (100 \AA) In containing layer.

STANDARD HEMT EPI-LAYER :	PSEUDOMORPHIC HEMT EPI-LAYER :
500 Å GaAs 1.5×10^{18}	500 Å GaAs Nd = 3×10^{18}
230 Å GaAlAs 28% undoped	230 Å GaAlAs 22% undoped
250 Å GaAlAs 28% Nd = 1.8×10^{18}	250 Å GaAlAs 22% Nd = 2×10^{18}
20 Å GaAlAs 28% undoped	30 Å GaAlAs 22% undoped
	100 Å GaInAs 20% undoped
5000 Å GaAs undoped	3500 Å GaAs undoped
SUBSTRATE	SUBSTRATE

Fig. 13: Conventional and pseudomorphic (PM) HEMT epitaxial layer structures.

First it has been shown that high purity material - which is a prerequisite - is reproducibly grown in the prototype system; the record mobilities published in 1988 ($\sim 700,000 \text{ cm}^2/\text{V.s}$ at 4°K) have been obtained again after more than 400 runs.

For conventional HEMT's, the 2 DEG structures exhibit typically $6000 \text{ cm}^2/\text{V.s}$ at a sheet electron concentration of $1.3 \cdot 10^{12}/\text{cm}^2$. Table II shows the sheet resistances obtained over five runs, measured on five points per wafer by a Tencor apparatus for contactless measurements. The wafers are then processed at LEP by a conventional HEMT process (0.5 μm gatelength), and high frequency measurements done by cascade microprobe. Fig. 13 shows the extrinsic unity current gain cut-off frequency. Although the dispersion of this last one is small, taking into account Table II, it may be stated that it is mainly due to the processing and that the uniformity and reproducibility of the epitaxial materials would allow much narrower histograms.

wafer run	1	2	3	4	5	6	7
1	362	359	370	371	366	363	364
2	370	370	374	370	370	373	376
3	369	371	371	371	376	372	369
4	360	359	363	358	364	362	361
5	358	357	354	355	356	358	357
6	368	373	376	372	371	372	372
7	369	368	372	367	365	368	370
8	370	371	375	371	373	369	369

Table II: Sheet resistance of HEMT structures for 8 runs of 7 wafers. Each value is the average over 5 points per wafer.

The pseudomorphic HEMT structure is depicted in fig. 13. It exhibits high sheet electron concentrations of typically $2.2 \cdot 10^{12}/\text{cm}^2$. A series of 25 successive growth runs

of these structures had been made, with 7 two inch wafers for each. One of the layers has been systematically characterized by various techniques, and the corresponding histograms show always very low dispersion (see fig. 14). Some of the wafers have been processed into HEMT's with $0.5\ \mu\text{m}$ to $0.25\ \mu\text{m}$ gate length. Table III shows some of the microwave transistor characteristics and fig. 15 the histogram of the current gain cut-off frequency which are state of the art. More statistical data will be available in a later stage of the project.

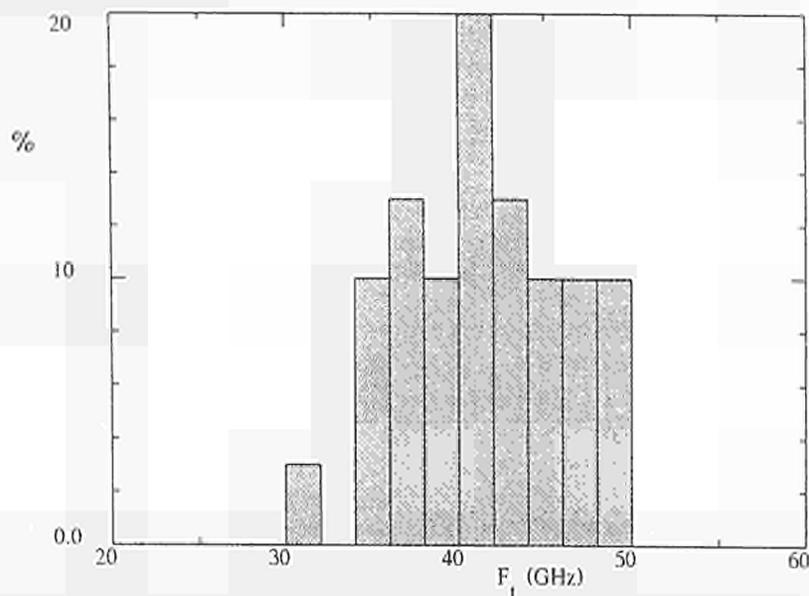


Fig. 14: Histogram of the extrinsic unity current gain cut-off frequencies (F_t) observed on $0.5\ \mu\text{m}$ gate length HEMT's.

Performances hyper optimales			
Parametres	Unités	opt. moy.	moy. opt.
F_t ext	GHz	53.7	53.8
F_t int	GHz	64.1	64.5
V_{gs}	V	0.07	0.07
I_{ds}	mA/mm	249	251
G_m int	mS/mm	477	477
C_{gs}	pF/mm	1.19	1.18
C_{gd}	pF/mm	0.23	0.23
G_d	mS/mm	23	39
τ_d	ps	0.36	0.34

Table III: Characteristics of a first series of $0.25\ \mu\text{m}$ (PM) HEMT's fabricated on Planet grown epitaxial layers (mean values).

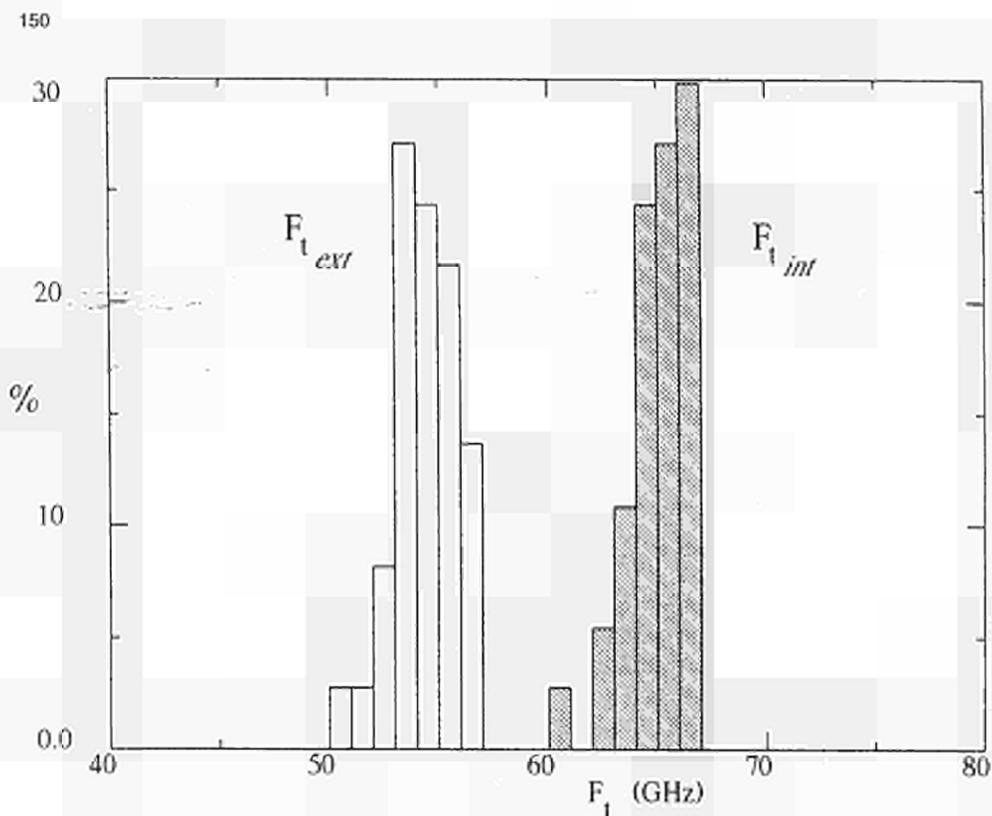


Fig. 15: Histogram of the cut-off frequencies measured on $0.2\ \mu\text{m}$ gate lengths (PM) HEMT's.

5. Evaluation of the reactor in opto-electronics

The feasibility demonstration of short wavelength lasers has been started in the second year in the new prototype reactor.

The first prerequisite is the purity of the active layers, the non radiative recombination time measured on the layer grown in this reactor indicates that this material is of excellent quality for laser structures. Subsequently, double quantum well laser structures have been grown for 850 nm emission. The very low threshold current density ($400\ \text{A}/\text{cm}^2$) measured on the first broad area lasers made from this material augurs well of the results we expect to demonstrate further the capacity of the Planetary Reactor to supply mass laser manufacturing.

6. Contribution to European Electronics

The following relevant contributions may be pointed out so far:

- The "Planet" type high throughput, industrial MOVPE equipment is already commercially available from AIXTRON. The system, based on these developments is called AIX 2000. AIXTRON is the world leading manufacturer of epitaxial systems with a world market share of approximate 35 %, and an European one of approximately 75 % with growing sales of production systems to the semiconductor industry. The AIX 2000 with its performances is undoubtedly ahead of the competitors, it has already been sold several times and show much promise to increase further AIXTRON

market share. It is successfully in operation for both GaAs/AlGaAs and GaAs/GaN.

- b) A state of the art pseudomorphic HEMT foundry process, based on Planet type MOVPE material is currently under development at Philips Microwave Limeil (PML) and will be commercially available in 1992. PML the really open GaAs foundry and ASIC Design Center now offers 5 fully released and calibrated MESFET processes. The new PML HEMT process will put PML in a leading position in Europe by offering manufacturing capacity for the most advanced microwave ASIC's.
- c) The new industrial unit "Philips Optoelectronic Centre" intends to industrialize mass production of lasers, based on the Planet type MOVPE materials. The advantage of high throughput and high uniformity will certainly allow to increase the competitiveness of POC, especially in the field of visible lasers where they are already in a leading position in R and D.
- d) POLYFLOW will make commercially available a comprehensive software package to be used as a general tool for predicting the flow structures, the temperatures, the concentration of species and the growth rate profiles of similar systems.
- e) Furthermore, it is also envisaged to make the supply of high quality, standard HEMT, LM HEMT or epitaxial material produced in Planet type reactors commercially available from PML.

CONCLUSIONS

Within the framework of this ESPRIT programme, it has been demonstrated that the new Planet concept of our MOVPE system is well fitted to supply the sophisticated epitaxial heterostructures required for the developing applications in micro and optoelectronics.

The availability of the equipment, the development of the related know-how and applications contribute already significantly to the European electronics.

REFERENCES

1. F. NAKAMURA and K. KAWAI, J. Cryst. Growth 93 (1988) 292-300
2. H. ITOH, T. TANAKA, T. OHORI, K. KASAI, E. MITANI, M. SUSUKI, J. KOMENO, Jap. J. Appl. Phys. n 10 (1989) L1693-L1695
3. G.S. TOMPA, M.A. McKEE, C. BECKAM, P.A. ZAWADZKI, J.M. COLABELLA, P.D. REINERT, K. CAPUDER, R.A. STALL and P.E. NORRIS, J. Cryst. Growth 93 (1988) 220-227
4. P.M. FRIJLINK, J. Cryst. Growth 93 (1988) 207-215
5. H.P.M. AMBROSIUS, R.W.M. LINDERS, C. WAUCQUEZ and J.M. MARCHAL, 5th Appl. Phys. Conf. 1991

DEVELOPMENT OF A LOW STRESS HIGH PINCOUNT PLASTIC PACKAGE FOR HIGH RELIABILITY CMOS ASIC's

A. Claes
Mietec Alcatel
Westerring 15
B-9700 Oudenaarde

G. Kelly
NMRC University College
Lee Maltings, Prospect Row
Cork, Ireland

J. Exposito
SGS-Thomson Microelectronics
Avenue des Martyrs, B.P. 217
F-38019 Grenoble

G. Neef
SEL Alcatel
Lorenzstrasse 10
D-7000 Stuttgart

K. Zachariassen
ElektronikCentralen
Venlighedsvej 4
DK-2970 Hoersholm

SUMMARY

This paper highlights the approach used for the development and evaluation of a low stress high pincount PQFP package for high reliability CMOS ASIC applications in the field of telecommunications, industrial control, automotive and computer electronics.

Reliable die attach materials and molding compounds for a low stress PQFP package are selected as the result of intensive material evaluation and process optimization. Mechanical stress simulations have been carried out to explain and confirm the mechanical stress measurements performed with the use of specially designed stress sensitive test devices. Reliability tests based on accelerated testing techniques have been performed using dedicated reliability test devices in order to assess the reliability performance of the developed PQFP package.

1. INTRODUCTION

Integrated circuit packages are designed to provide an efficient mechanical and electrical interface between the chip and the circuit board, as well as providing adequate heat dissipation and protecting the chip against failure inducing contaminants.

Ceramic packaging is the most reliable packaging technique and is well suited for high performance high reliability IC's, but it is the most expensive packaging technique. Due to its highly automated process and its low cost materials, plastic packaging is the most widely used packaging technique for low pin count integrated circuits. However, in recent years the tendency towards increased complexity in VLSI designs has resulted in IC's with higher pin counts, larger die area's, higher power dissipation and higher operation speeds. Then most types of plastic packages are not suitable for these applications. One of the major problems to face is the thermomechanical stress induced by differences between the thermal coefficient of expansion of the silicon and the plastic encapsulation material.

The main objective of the ESPRIT 5033 (PLASIC) project on the performance and reliability of plastic encapsulated CMOS ASIC's is to obtain well founded answers and guidelines on how to build and evaluate a reliable plastic package for large die size CMOS ASIC's within the field of high reliability applications as the ones which are encountered in telecommunications, industrial control, automotive and computer electronics.

Throughout this project the main failure mechanisms due to plastic packaging will be identified, especially the ones which are induced by the interactions of complex high reliability devices of large die dimensions with a high pin count plastic package used for surface mounting on printed circuit boards.

Finally, at the end of the 3-year project a methodology will be defined for the qualification of high reliability plastic packaged CMOS ASIC's using modified test methods and faster test sequences than the ones which are presently in use. This will lead to the definition and proposal of a new standard for accelerated testing.

Both users and producers of plastic packaged devices are represented in this consortium. This will help both parties to find a common agreement about a major part of the reliability hazards related to the use of plastic packages in the field of complex high reliability VLSI ASIC components and therefore to understand each other in a better way since they will use the same evaluation criteria and methods of accelerated testing for assessing the reliability of plastic packaged devices.

2. PROJECT STRATEGY

The ESPRIT 5033 (PLASIC) project consortium selected a 144 pins plastic quad flat package (PQFP) as an example of a high pin count plastic package used for surface mounting on printed circuit boards.

The project has started with an intensive material study and process optimization in order to develop a low stress PQFP package.

Due to the limited understanding of the failure mechanisms involved during accelerated stress tests, it has been found necessary to develop common and well proven acceleration test methods in order to evaluate the reliability of plastic packaged complex CMOS ASIC's.

Special focus is also put on the development of suitable thermo-mechanical modelling tools for the evaluation of the thermal and mechanical stress induced by the plastic encapsulation on complex high reliability ASIC devices.

The influence of the plastic package on the performance of devices and of analog and digital building blocks will be studied and explained with the information gained from the stress measurements. The electrical and thermal properties and frequency behaviour of the package will be evaluated.

A demonstrator circuit will be redesigned using the design rules derived from the measurements on test chips. It will be processed and packaged in a PQFP package and submitted to the developed qualification procedure.

This paper highlights the approach used for the development and evaluation of a low stress PQFP package consisting of the following activities: material evaluation and process optimization, mechanical stress measurements, mechanical stress simulation and accelerated testing of the package. These activities are briefly described in the following paragraphs.

3. DEVELOPMENT AND EVALUATION OF A LOW STRESS PQFP PACKAGE.

3.1. Materials selection and process optimization

A. Selection of die attach material

A survey of existing die attach materials was performed according to physical, mechanical and process criteria. The prospection is limited to the epoxy materials which have a lower curing temperature and are easier to process than polyimide. From information issued by suppliers a selection of nine materials satisfying the following characteristics has been made: epoxy type resin, one component silver filled resin, low elasticity modulus, very low ionic impurities content (Cl^- , Na^+ , K^+), non-solvent content, viscosity and thixotropy in the desired range and low outgassing after cure.

The selected materials are submitted to further evaluation using four comparison criteria. As summarized in Table 1 appropriate test vehicles are used for each criterion.

Comparison criterion	Test vehicle
Thermally induced stress warping measurements	30x5x0.5 mm ³ silicon chip bonded on 33x8x0.1 mm ³ copper frame
Shear strength at 25°C and 150°C	Silicon chip on silver plated Alloy 42 frame
Thermal cycling fatigue	9x9 mm ² silicon bonded on copper frame
Outgassing	Weight loss during and after polymerization

Table 1: Summary of the test vehicles used to evaluate the different die attach materials for the different comparison criteria.

For the first two criteria a test vehicle with a copper frame is used in order to achieve the comparison in the most stressful configuration.

Die warping measurements and shear strength measurements are used to subdivide the die attach materials into two classes: soft and hard adhesives. The ranges according to both classes are given in Table 2. The thermal cycling fatigue and the outgassing properties are used to select one material out of each class.

Die attach material	Die warping measurement	Shear strength
Classic (hard) adhesive	> 150 μm	> 2.5 kg/mm ²
Soft adhesive	< 75 μm	< 1.5 kg/mm ²

Table 2: Classification of the die attach materials based on die warping measurements and shear strength measurements.

For both materials a die attach process optimization has been carried out. The selection of the nozzle type and the dispensing and bonding parameters was possible making use of the following methods:

- X-ray microscopy and die breaking off to check voids and incomplete filling
- microsectioning to measure the glue thickness.

Adhesive curing temperature and duration has been optimized with respect to shear strength and thermal stress (warping measurements).

B. Selection of molding compound materials

A molding compound for high pincount and large die packages for surface mount technology should satisfy special mechanical stress related requirements in addition to the general requirements for reliable plastic packages. The general requirements are low ionic content, good thermal conductivity, low water absorption, good adhesion with lead-frame and die and finally, good moldability. The stress related requirements, expressed in thermal mismatch and popcorn effect, are related to the Thermal Expansion Coefficient (TEC) and the flexural modulus (E) of the molding compound.

Matching of the TEC value of each package component is necessary in order to minimize the internal stress in the package giving rise to package failures. Lowering the TEC value of the molding compound together with the E modulus minimizes mechanical stresses.

However, to avoid popcorn failures, a higher TEC is desirable together with a higher glass transition temperature (T_g) and higher E modulus. The popcorn failures refer to the package cracking caused during reflow soldering by the evaporation of moisture trapped at the delaminated interfaces between resin and molded parts.

Based on suppliers information on TEC, T_g, E, flexural strength (S) and ionic content, seven molding materials have been selected for further evaluation.

A first step in this evaluation is a physico-chemical analysis, consisting of checking the suppliers data on TEC, T_g, E modulus, ionic content, water absorption, spiral flow, flash test and gel time. A range of the major molding compound properties has been defined according to the needs for PQFP 160 package.

- T_g : 160°C
- TEC : 15-17 ppm/°C
- E : 1150 kg/cm²
- Na⁺ ionic content < 1 ppm
- Cl⁻ ionic content < 5 ppm.

A second step of the evaluation consisted of checking the moldability through voids, flashes, incompletes, sticking, cracks and wire sweep observed on molded PQFP compounds.

Finally the following reliability tests were performed on three retained materials to select two molding compounds for further process optimization :

- Water absorption at 80°C/100% Relative Humidity (RH)
- Dye penetrant at 20 bar/4h
- Pressure Cooker Test (PCT) at 121°C/2atm/100%RH
- Temperature Humidity Bias Test (THB) at 85°C/85%RH
- Thermal cycling (TC) at 0°C/125°C

3.2. Mechanical stress evaluation

Two different methods [1] and test dice are used to evaluate the thermomechanical stress induced by the plastic package. The first method uses a test die with structures sensitive to passivation cracks and metal displacement. The second method uses piezoresistive stress sensors. In this paragraph both evaluation methods are compared.

A. Metal displacement measurement

The test die is a two layer device consisting of Al patterns with a thickness of 2.4 μm covered by a passivation sandwich layer of 0.4 μm silicon oxide and 0.8 μm silicon

nitride. Because the passivation thickness is less than the metal thickness this test die is very susceptible to passivation cracks induced by mechanical stress [2]. Two test devices, one including large metal patterns (SD1) and one with a parallel striplines network (SD3), are used for visual inspection after chemical opening of the package. Figure 1 gives a lay-out of the test devices. The test die SD1 is used for failure analysis while SD3 is used to evaluate the effect of the die size.

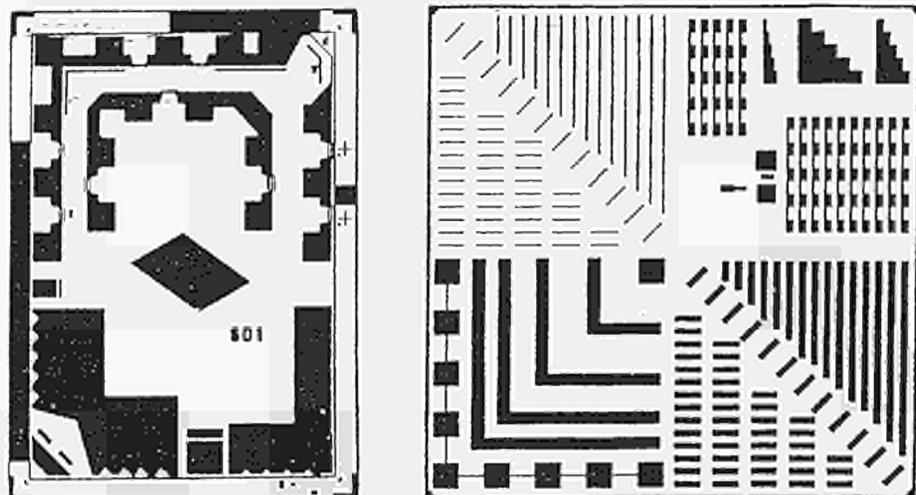


Figure 1: Layout of the metal displacement test devices SD1 and SD3.

Modules of 30 dice ($8.85 \times 7.8 \text{ mm}^2$) are assembled in a PQFP package with the two selected molding compounds and evaluated after molding and after application of different numbers of thermal cycles ($-55^\circ\text{C}/+150^\circ\text{C}$). Table 3 gives an overview of the different failures and the location of the failures observed during the cycling experiment.

The corner of the die is first affected by passivation cracks and metal displacement, which means that the die corner is the highest stress region. This stress acts also on the edge of the die. The centre of the die is not affected. The forces causing this metal shift have to be shear forces acting on the die surface forcing the metal towards the centre of the chip.

Since the passivation cracks and the metal displacements are caused by shear forces acting on the metal and the passivation, these experiments permit to identify the high stress regions on a die packaged in a PQFP package.

Test	Failure	Location
After molding	Little cracks in passivation	Die corner
40 thermal cycles	Passivation cracks	Die corner + edge
	Small metal displacement	Die corner
200 thermal cycles	Increase of crack density	Die corner + edge
	Increase of crack dimensions	
	Metal displacement (largest for molding compound B)	Die corner + edge Maximum shift in die corner

Table 3: Overview of the different failures and the location of the failures during the cycling experiment

In order to check the effect of the die size, modules of different numbers of the test die are assembled in a PQFP package and submitted to 400 thermal cycles ($-55^{\circ}\text{C}/+150^{\circ}\text{C}$). The metal displacement distribution on the die surface, issued from these observations, are summarized in figure 2. The area where a metal shift is observed (expressed in distance from the edge of the die) increases with increasing die size. The metal shift amplitude is larger in the die corner than at the die edge. Figure 3 gives a plot with a quantification of the metal displacement over the test die.

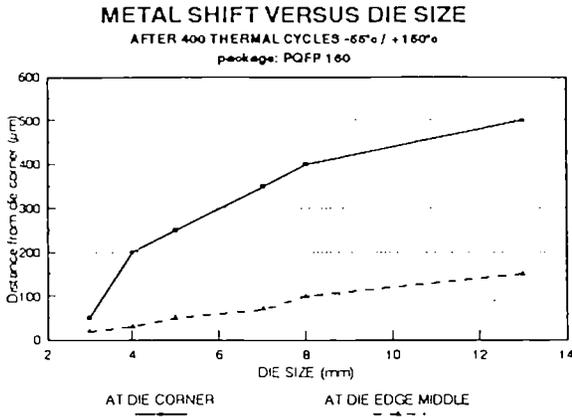


Figure 2: Metal shift area at die corner and die edge for different die sizes.

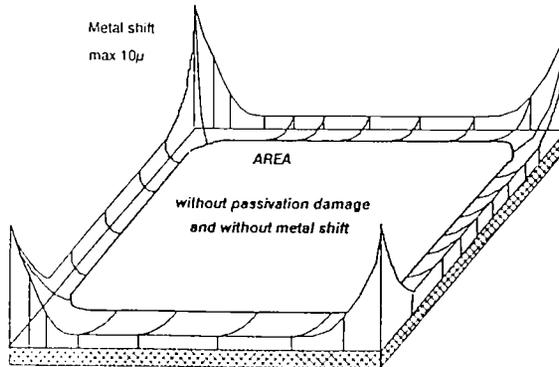


Figure 3: Metal displacement quantification over the test chip.

B. Piezoresistive stress measurement

With the metal displacement stress die it is possible to identify the high shear stress regions in a PQFP package. However, no detailed information about the various stress components in the package can be obtained. A test chip with piezoresistive diffused resistors in silicon is used to quantify the different stress components in the package. The piezoresistive elements are low doped n-type diffused resistors of about $1\text{ k}\Omega$. Four resistors are combined to form a unit that allows the calculation of the components σ_x , σ_y and τ_{xy} on the silicon plane where the unit is located. A diode for silicon temperature

evaluation is also provided on the test die. Figure 4 gives a schematic view of the piezoresistive die. The die size is $2.9 \times 2 \text{ mm}^2$.

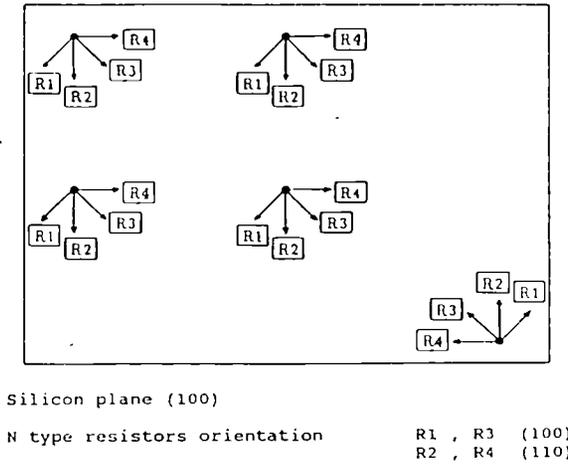


Figure 4: Schematic view of the piezoresistive test device.

Modules of 12 test dice are assembled in a PQFP package with the two selected molding compounds. The results are summarized in Figures 5 and 6 where the different stress components are given for different locations on the die surface. Compressive stresses parallel to the die edges form the largest components. The largest σ_x and σ_y occur at the centre of the x and y edges respectively. Shear stress in the surface plane τ_{xy} maximizes at the corners and decreases towards zero along the centre line of the die.

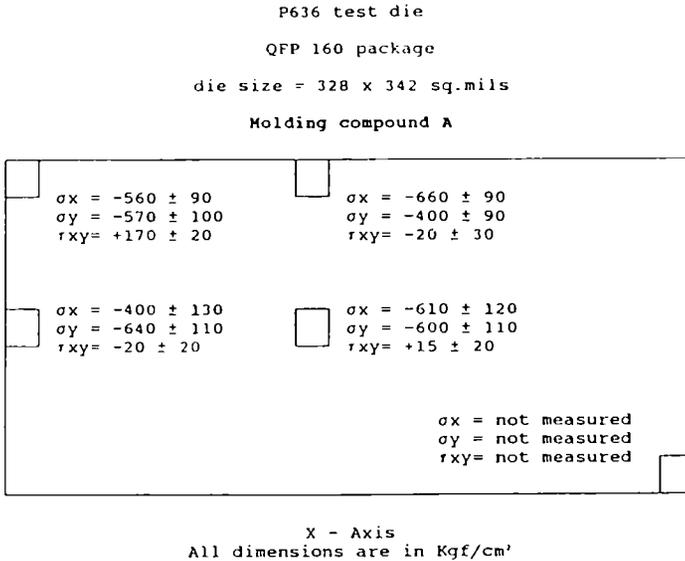


Figure 5: Stress components for molding compound A.

Molding compound B shows a larger shear stress at the die corner, which is in agreement with the metal displacement observations discussed in the previous paragraph.

Modules of different numbers of test dice are assembled in PQFP packages with molding compound B. Figure 7 gives compressive and shear stress variations versus die size. The shear stress at die corner increases with increasing die size in agreement with metal shift area variation versus die size given in figure 2.

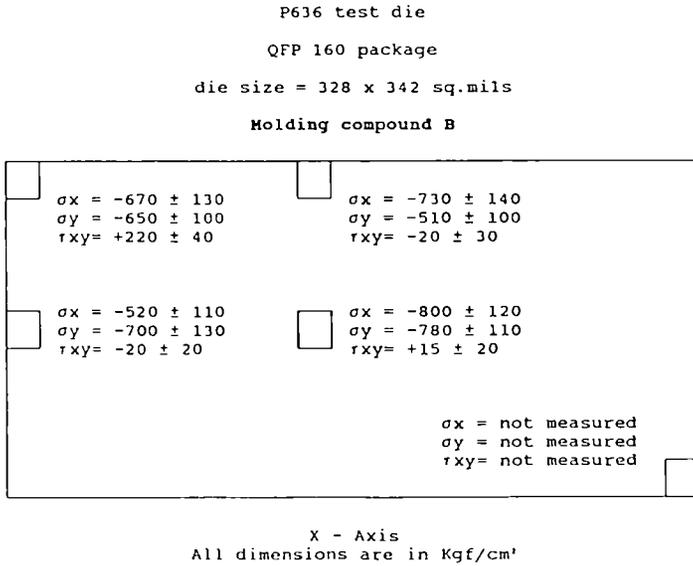


Figure 6: Stress components for molding compound B.

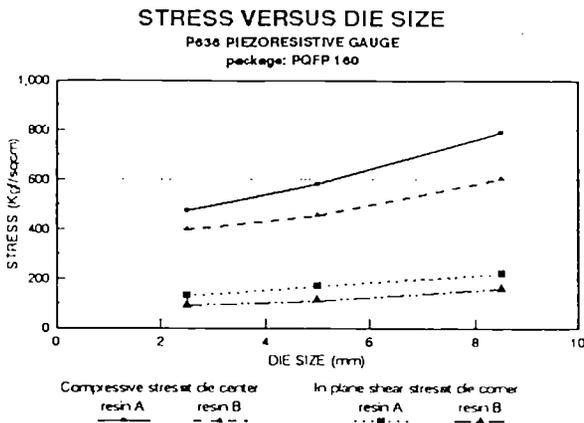


Figure 7: Mechanical stress components for different die sizes.

3.3. Simulation of the mechanical stress

The three dimensional finite element (F.E.) analysis of a PQFP 160 was undertaken on a F.E. code NISA to more accurately determine the shear stress distribution on the surface die. The results of the simulation give an insight into the relationship between out of plane shear stress on die surface and observed metal shift patterns.

Three shear stress components act on the surface of the die: namely an in-plane component τ_{zx} and two out-of-plane components τ_{xy} and τ_{yz} . The F.E. model results indicate that the in-plane shear stress component τ_{zx} is concentrated at the corners and decays rapidly towards the centre. This trend is shown in figure 8 and compares well with strain gauge measurements, as shown on figure 6. In addition, two out of plane shear stress components τ_{xy} , τ_{yz} of similar magnitude to τ_{zx} , act along the edges of the die and decay rapidly towards the die centre, as shown in figure 9. Furthermore the sign/direction of these shear components is such that these forces act towards the centre, and are perpendicular to the edge of the die. The edges of the die are subjected to locally very high forces throughout their length.

It is proposed that the inward movement of the metal lines along the die edge is caused either in part or completely by the out-of-plane shear stresses described earlier. These stresses are essentially surface stresses locally high at the edges of the die, whose line of action is inwards towards the centre of the die and which are concentrated in a thin region/strip close to the edges of the die. The two distributions of shear stress, figures 8 and 9 and metal shift figure 3, are very similar in appearance and outline.

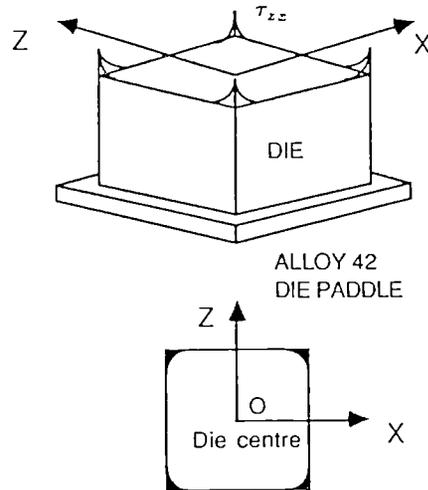


Figure 8: Region of high in-plane shear stress acting on the die surface.

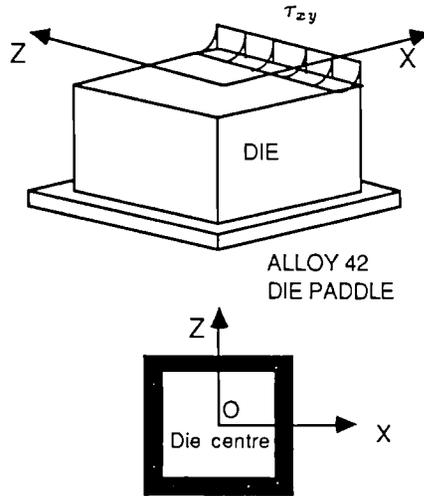


Figure 9: Region of high out-of-plane shear stress acting on the die surface.

Reliability evaluation of the package

Since a plastic package is not impermeable to moisture the devices have to be submitted to accelerated humidity tests such as the HAST tests (Highly Accelerated Stress Test carried out at 125°C/86%RH or 135°C/86%RH). Because of the thermal mismatch between the thermal expansion coefficients of the different components of the package, the devices are also submitted to temperature shock and cycling tests to check the reliability .

A dedicated testchip for these reliability tests of the package has been designed, consisting of double metal meanders which can be stressed at different voltages during HAST tests. At the corners of the chip different metal patterns are placed to study the influence of the layout on passivation cracks and metal displacement. A schematic view of this test chip is given in Figure 10. The processing and packaging details are given below:

- Chip dimensions : 4 test dice of 4 x 4 mm² in 1 PQFP package
- Passivation : 1.1μ m silicon nitride
- Metallisation : 0.8 μm Al/Si/Cu
- Leadframe : Alloy 42
- Die attach and molding compound : 4 combinations of the selected materials

In contrast with the metal displacement testchip the passivation thickness is larger than the metallisation thickness as required for reliable plastic packaged chips.

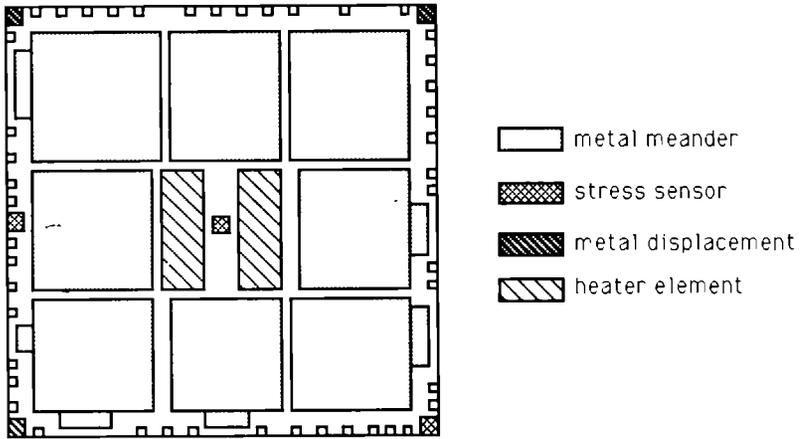


Figure 10: Schematic view of the reliability test chip.

A. Humidity tests

The packaged die was submitted to a variety of HAST tests with different cycling tests as preconditioning. Table 4 gives an overview of the different tests applied to the devices. A failure is defined as an interruption of a metal meander. Molding compound A is submitted to different HAST tests. Figure 11 shows that molding compound B successfully passes the most severe HAST condition (first failures at 1277 hrs HAST).

tests number	molding compound	preconditioning thermal cycling temp.	HAST conditions temp/RH/bias
1	A	-55°C/125°C	125°C/85%/10V
2	A	-55°C/125°C	135°C/85%/10V
3	A	-65°C/130°C	135°C/85%/10V
4	B	-65°C/130°C	135°C/85%/10V

Table 4: Overview of the different HAST conditions for different molding compounds.

Cumulative % failures for different HAST tests

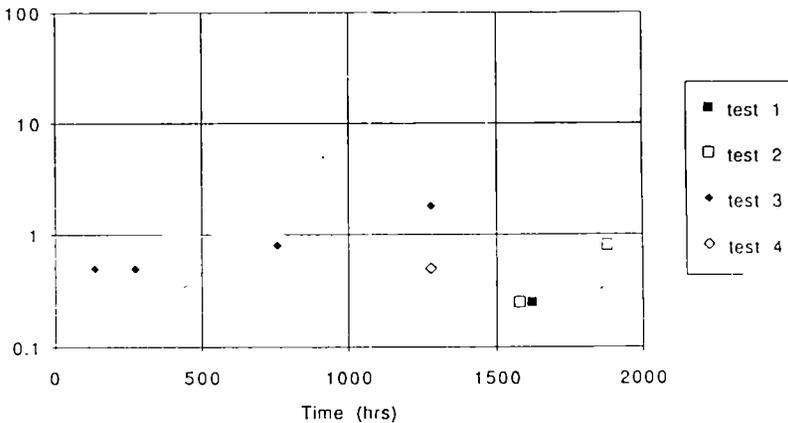


Figure 11: Cumulative percentage of failures for different HAST tests.

A. Shock and cycling tests

The temperature cycling test between -65°C and 150°C was selected as the most severe shock or cycle test. However, no meander resistance changes, no passivation cracks and no metal displacement could be observed after 1000 cycles. After different numbers of cycles, devices have been taken out for detailed failure analysis. The devices are subjected to high resolution real time X-ray inspection to check voids in the die attach material. Some very small voids in the hard epoxy die attach are observed. Temperature cycling has no influence on this failure. The devices are further analyzed with acoustic scanning microscopy using an Olympus UH3 SAM instrumentation (at 30 MHz) or a P.SCAN instrumentation (at 10 MHz). Die separation or delamination can be observed on a limited number of devices. Figure 12 gives an acoustic scanning micrograph of a device with corner delamination. Extensive cycling between -65°C and 150°C first increases separation/delamination at the corner and later at the edge of the die (Figure 13).

It is reasonable to suppose that the delamination or separation is caused by shear stress acting on the surface of the die since stress measurements and simulations proved that the shear stress reaches its maximum value at the die corners.

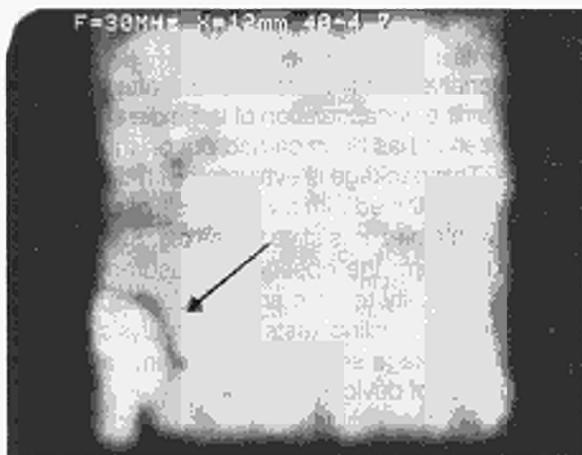


Figure 12: SAM micrograph of a device with corner delamination.

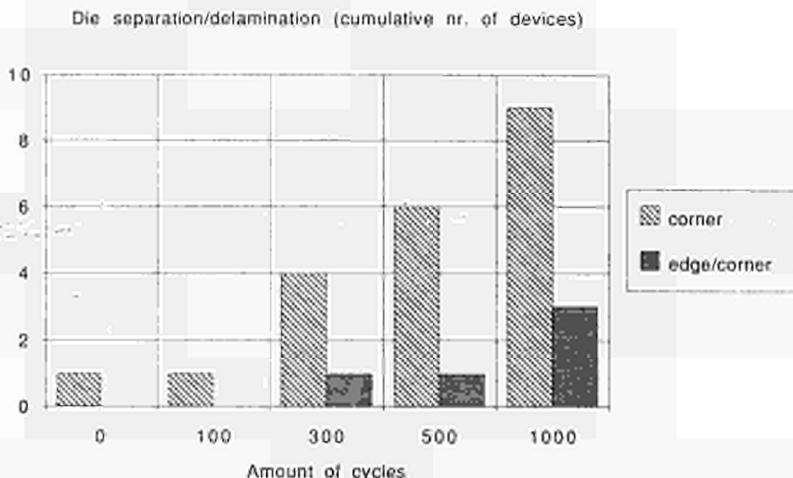


Figure 13: Cumulative amount of die separation/delamination at the die edge and corner for different numbers of temperature cycles.

4. CONCLUSIONS

Through intensive material evaluation, reliable die attach materials and molding compounds are selected for the encapsulation of large die size CMOS ASIC's in PQFP packages. Process optimization has been carried out on these materials, leading to a low stress PQFP package. This package is evaluated with different test dice. The metal displacement measurements obtained with a very sensitive test die is in agreement with the stress measurements with a piezoresistive stress die. Stress simulations have been carried out to explain and confirm the observed measurements. Reliability tests are performed on a dedicated reliability test die and the developed package successfully passes HAST and temperature cycling tests. The analysis with acoustic scanning microscopy shows that die package separation or delamination mainly occurs at the die corner on a limited number of devices.

5. ACKNOWLEDGEMENTS

The authors would like to thank the colleagues of the participating companies of the ESPRIT 5033 (PLASIC) project for their contribution to the investigations.

6. REFERENCES

- [1] R. Tiziani et al.: "Plastic Package Reliability Study by means of integrated test structures.", 8th Proceedings IEMT (IEEE), 1990, Baveno Italy.
- [2] A. Hente et al.: "Top Layer and Assembly Route for Optimal Corrosion Performance of Plastic Encapsulated Devices.", 4th Int. Conf. Qual. in Electr. Comp. '89, Bordeaux France.

RAPID THERMAL PROCESSING EQUIPMENT DESIGN AND INTEGRATION FOR A CLUSTER TOOL MODULE

T. THEILER
A.S.T. Elektronik GmbH
Helmholtzstr. 20
D-7900 Ulm
Tel.: (49) 731 51074
FAX: (49) 731 54258

SUMMARY

The major target of this work is to combine and integrate process steps which are required for submicron ASIC and memory device fabrication in CMOS or BICMOS technology in an automated manufacturing system, where the wafer is transported under vacuum conditions from one process chamber to the next. A.S.T. Elektronik GmbH has built up and integrated a Rapid Thermal Process chamber into the clusterline concept CLC 9000 of Balzers. RTP offers a cold wall system for high temperature wafer processing. In contrast to furnace annealing the wafer itself is heated very quickly to a high temperature by radiation. This minimizes contamination. As a cold-wall system RTP collects the advantages of vacuum and low pressure technology. In highly integrated circuits manufacturing in Si technology is often a high temperature and atmospheric pressure process step combined with a low temperature and low pressure step being sensitive to ambient and surface contamination. Integration of the process sequence into a multichamber high vacuum system can avoid ambient and surface contamination problems. This would be helpful for various processes like Ti-silicidation or Al-reflow. The RTP-cell developed by AST and now being presented could be used on a modified basis also for other applications. A joint cooperation with Balzers, Plasmos and Siemens will enable AST to reach new markets and participate in this product structure.

1. INTRODUCTION AND OVERVIEW

A very important achievement in IC technology is the silicide (self aligned silicide) process which serves to reduce the number of masking steps in IC-technology. Various publications confirm, that the Ti-silicide process is generally viewed as a very promising process sequence for integrated processing^{1, 2, 3}. Preclean with soft sputter etch is by many researchers seen as one of the key points for a success of this process sequence. In this process a metal layer is deposited over a MOS-structure and then made to react with the exposed Si areas of the source/drain region and the poly-Si in the gate to form the silicide. In the silicide technology the polysilicon gate is patterned before the silicide formation. An oxide layer is deposited by CVD and anisotropically etched down leaving insulating sidewall spacers beside the poly-Si gate. These spacers have two purposes: they serve as masks during the ion implantation of source/drain areas but also prevent the gate and source/drain regions from being electrically short-circuited. The metal to form the silicide is then deposited and reacted with the silicon. The remaining metal is etched away by using a selective etch that does not react with the silicide. Finally the device is glass passivated and contacts are made.

Within the ESPRIT 5041 project the process modules for three process configurations are developed and integrated around a central handling transport system already built up by one of the partners. In this manner up to nine different process chambers can be brought together into one production line, with the advantage to transport wafers automatically from one station to the next under ultraclean high vacuum conditions. The three process configurations are:

- Advanced multimetal sputtering
- Tungsten CVD
- Planarized intermetal dielectric deposition

Within the workpackage "Advanced Multimetal Sputtering" three process stations are required:

- A soft sputter etch station for substrate cleaning
- A metal sputtering station for titan deposition
- A rapid thermal processing station working at different pressures with defined atmosphere and temperature for annealing.

The design, development and building up of this module is the contributing part of A.S.T. Elektronik GmbH.

2. THE ROLE OF RTP IN MANUFACTURING PROCESSES

The development and manufacturing of denser circuits with 0.5 micron geometrics requires a new type of process equipment for film deposition and etching. The generic cluster tool concept has the following features:

- minimum of manpower for operation
- extremely low wafer surface contamination and thus maximal gain
- multistep metal deposition
- in situ cleaning of etch and deposition chambers

Experts expect around 1994 a 20 percent use of cluster tools for volume manufacturing.

Furthermore the fast development of new processes and their quick integration into manufacturing lines asks for an open system architecture which allows the fast and reliable integration of new processes and/or new process modules into an existing and proven basic equipment concept. One important part of such a cluster line production sequence is a rapid thermal processing chamber.

Several process sequences involving rapid thermal processing systems have been proposed by the IC-manufacturers:

- The titanium salicidation is one possible process sequence where a clusterline concept could be more useful than stand alone applications. The steps are:
 1. Soft etch (ca. 5 nm etching) with bias up to 100 V for 30 seconds
 2. Titanium deposition (60 nm thickness in 30 seconds)
 3. First rapid thermal annealing at 650°C - 700°C, 30 seconds with 1 Torr N₂ gas flow, ramp down to room temperature

These processes can be done in a triple module station of the cluster line

4. selective Ti(N) etch wet outside the cluster line
5. 2nd RTP in a separate stand alone system after wet chemical removal of nonreacted Ti anneal time: 10 sec in N₂ at 800°C - 900°C; ramping: 50°C/sec in combination with BPSG reflow

- For sputter processes a sequence could be:
 1. Soft etch removal of native oxide on metal 1 (ca. 3 nm)
 2. Reactive sputtering e.g. TiN 100 nm
 3. Rapid thermal stuffing (structural changes of TiN) at 600°C - 700°C
 4. Standard deposition of Al with 1-2% Si or / and Cu (ca. 1 µm)
 - (5.) Rapid thermal annealing at 420°C for sintering the contact resistance
 - (6.) Photolithographic process outside the cluster line

- Another sequence where a clustering of process chambers becomes necessary is Al-reflow. Thereby aluminium is sputtered to the surface of Si in order to get good electric contacts. But the sputtering process alone does not yield homogeneous layers in contact windows. As a second step rapid thermal annealing (at 480°C - 500°C) becomes necessary for resistive homogeneous contacts. During both processes it is immediately necessary to avoid any kind of O₂ or H₂O contamination.

- In order to attain good ohmic contacts all sputtering processes in combination with rapid thermal processing could be useful in a clusterline concept.

AST has constructed a rapid thermal process chamber for use in a salicide process sequence. It was the intention to build up the module in such a way, that it can fulfil the various technical requirements of the proposed processes.

At the moment it is possible to implement this RTP-module into a single process chamber connected with the central handling system of Balzers, but with slight modifications the module can also be mounted in a triple station. The mechanical connections and electrical interfaces of our RTP-module are compatible with MESA (Modular Electronic Standard Architecture). Our rapid thermal processing module is designed to heat wafers up to 1000°C from one side. 35 kW halogen lamp power thereby is consumed. This will be the base for the further development of RTP-modules with two side heating possibilities, as required by the IC-manufacturers for oxidation processes. A high reflectivity of the process chamber walls together with the enormous radiation density available then, will allow temperatures much higher than required by any Si-wafer process and could yield to new and also exotic applications (for example high temperature material research applications).

The availability of RTP-modules for cluster environments will enable AST to reach new markets in the semiconductor industry and to participate in this project structure. The synergetic effects resulting from the teamwork with Balzers made it possible to develop a competitive rapid thermal processing module in comparatively short time.

3. PROBLEMS AND SOLUTIONS IN RTP WAFER PROCESSING

The close cooperation of A.S.T. Elektronik GmbH and Siemens within and beyond ESPRIT led to new results concerning inhomogeneities in ramp up and steady state heating and thus also to an improvement of process technology. Moreover, as a manufacturer of Rapid Thermal Systems A.S.T. Elektronik GmbH could get access to topic problems in semiconductor wafer treatment, knowledge which is absolutely necessary for staying competitive in future:

Three main sources of thermal gradients and inhomogeneities arising over the wafer can be located^{4,5,6,7,8}. They are of fundamental importance for the design of outstanding Rapid Thermal Systems:

1. The "photon box effect":

Monte-Carlo simulations by Kakoschke et al.⁴ (Siemens) showed that in the common arrangement of a reactor chamber (photon box) the probability of the photon absorption at the wafer edge is higher than in the middle of the wafer, resulting in an additional heating up of wafer regions close to the edge. This effect depends on wafer sizes and on given constructions of the reflector. It is more dominant the higher the difference between primary light (from the lamps including multiple reflections) and secondary light (heat radiation emitted from the wafer which is reabsorbed after single or multiple reflections) is, i. e. the higher the ramping rate is. This "photon box effect" is caused by the locally higher radiation intensity in chamber parts, where the photons do not pass the wafer and thus being many times reflected. In these regions a higher photon energy flux density is present during ramping up, i. e. as long as the wafer emission does not balance the absorbed radiation. As a consequence the homogeneity distortion while heating up changes with chamber geometries and wafer sizes.

This can be minimized by two ways: 1) By making the reactor dimensions very large compared to the wafer size or 2) by making the reactor dimensions as small as possible. But in order to remain variable with wafer sizes, a good possibility to reduce this effect is to optimize the heating program.

2. The "edge effect":

The edge effect is due to the fact that the surface-volume ratio at the wafer edge is higher than in the centre (the wafer does not have infinite dimensions). Because of the additional surface of the edge this part is emitting more radiation per time in steady state. The emitted intensity cannot be reabsorbed fully by the edge region again. As a consequence the edge of the wafer becomes colder than the centre.

3. The "pattern effect":

Suppose the following situation: A very thin, radiation absorbing layer is deposited on a SiO₂ structure on a wafer. Now, what is happening during a strong thermal ramp up phase? This layer reacts very quickly to the lamp power. It becomes very hot immediately at regions with the SiO₂ underneath due to the very low thermal conductivity of SiO₂.

This may result in an inhomogeneous heating of the surface layer, because layer regions deposited directly on Si can transport the thermal energy away much quicker. As a result this situation could lead to a damage of the surface layer, mainly in cases, where this layer is very thin.

One possibility to minimize pattern effects is optimization of temperature control. Avoiding quickly oscillating lamp powers, i.e. peaks in the ramp up rate is of fundamental importance. This is a great problem for RTP-producers working with arc lamps. As far as we know, A.S.T. Elektronik GmbH at the moment is the only RTP-manufacturing company, being able to take care of this situation. Our software features enable the operator to determine and limit the lamp power to exactly one fixed value per second. Because also our lamp power is protocolled as function of time, the operator is able to optimize recipes with sufficient smooth (and peakfree) lamp ramp up rates.

Additionally another fatal effect which could become dominant by use of arc lamps in RTP-systems is due to the very strong line spectrum of arc lamps. Interferences in optical thin films can heat up surface layer parts much stronger than halogen lamps

can do. Although these effects were not of primary interest for IC-manufacturers until now, in order to stay competitive also in future, A.S.T. Elektronik GmbH solved this problem early by use of halogen lamps and satisfying software features.

4. CONSTRUCTION OF A RAPID THERMAL PROCESS CHAMBER MODULE FOR A CLUSTERLINE

Figure 1 shows the heater block. The IR-lamps are radiating their energy through a quartz plate onto the wafer from one side. The temperature of the wafer can be determined either by thermocouples, or by pyrometers. Thermocouples are not able to reproduce steep ramp up rates because of their slow electrical response (> 1 sec) and therefore are mainly used in combination with a pyrometer to calibrate the temperature of the pyrometric output signal.

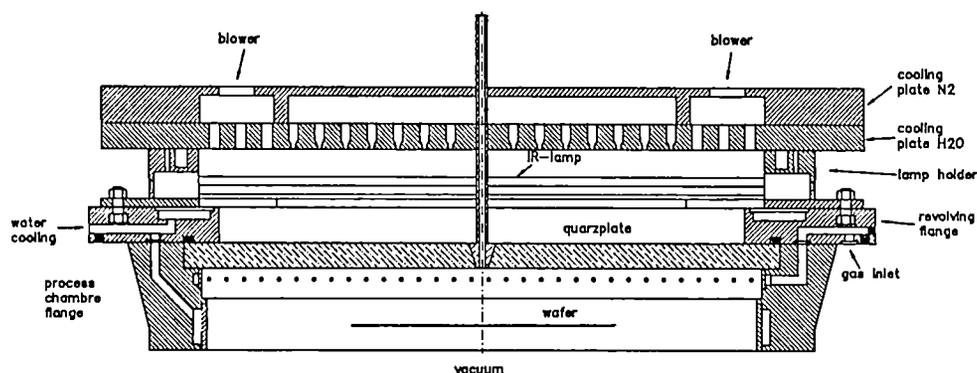


Fig. 1: RTP Module - Heating Unit

Measuring the emission of the heated wafer with a pyrometer has the benefit of recording quick temperature changes every 50 msec, but the emitted signal of the wafer depends strongly on surface characteristics and doping concentration. Therefore every type of wafer in principal needs its own calibration curve. A sapphire lightpipe can be used to guide the emitted radiation of the wafer out to the input of the pyrometer.

IR-lamps, lamp base and quartz plate are cooled with air or nitrogen via small holes in the lamp holding plate. Cooling of the quartz plate is necessary, because a significant part of the radiation intensity will be absorbed by quartz. As a consequence of the very small thermal conductivity of quartz glass, a thermal equilibrium of the quartz plate is

reached only after several minutes of processing time. This would mean, that the reproducibility of a process sequence could only be guaranteed after a considerable pre-heating time. A cooled quartz plate can minimize this pre-heating time.

Additionally the lamp holding plate is cooled by water to avoid heating up of the metal and the lamp bases, and also to make the gas cooling more efficient.

New lamp types have been selected. Halogen tubes of different lengths are used in order to illuminate a sufficient area of the cylindrically shaped reactor chamber. The lamp bank is rotatable in order to compensate for thermal variations over the wafer which can occur because of gas flow directions from one side to the other. The radiation in the other direction is optimized by a single lamp control system.

The single lamp control system could be developed within the project EP 5041 and can also be integrated into our stand alone applications. Based on this new technique in situ corrections are made for the dynamically occurring inhomogeneities during the ramp up phase, i.e. for the situation when the edge of the wafer becomes hotter than its center, and the case for constant radiation after several seconds. This can be managed for example by additional use and control of halogen single lamps arranged concentric between the heating tubes.

The process chamber is separated from the lamp housing by the quartz plate. The processing gas in the reaction chamber is distributed via small holes radially symmetric to the reflector. A concentric gas inlet together with a concentric pumping connection leads to most homogeneous gas distributions in cylindrical reactor chambers. Because the processing gas is not pumped off concentric, the lamp holder is also rotatable to enable adjustment of thermal gradients, which can arise due to an overall process gas flow direction. For the case that a sufficient homogeneous gas distribution cannot be obtained, a quartz process gas distributing plate can also be easily implemented within the reactor chamber.

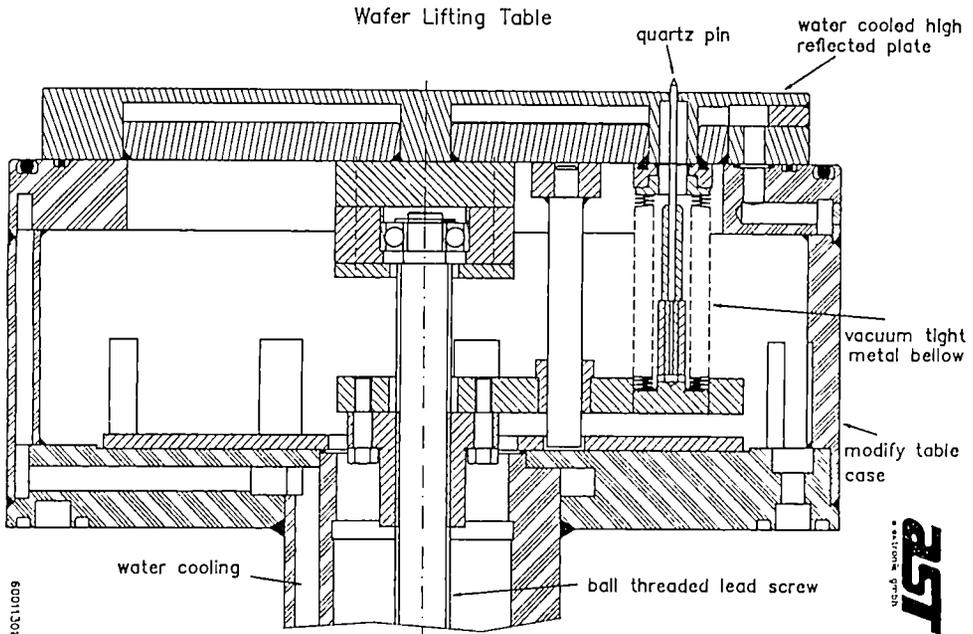


Fig. 2: Wafer Lifting Table for RTP-Module



Figure 2 shows the wafer lifting table. The wafer movement on the workholder is managed by a linear drive working in the table box. The linear drive moves three quartz pins up and down, which are for vacuum tightness, fixed in metal bellows. The temperature of the wafer can be measured from below by use of fibre optics in the ball threaded lead screw guiding the emitted light from the wafer backside to the pyrometer cell.

This option is useful in cases where reflectivity or emissivity of the wafer changes at constant temperature during processing.

In order to attain quick cooling times the wafer lifting table is water cooled. After processing, the linear drive lowers the wafer onto the table, and the wafer is rapidly cooled.

Afterwards the wafer lifting table is lowering, thereby opening the process chamber, so that the wafer can be transported by the fully automatic handling system.

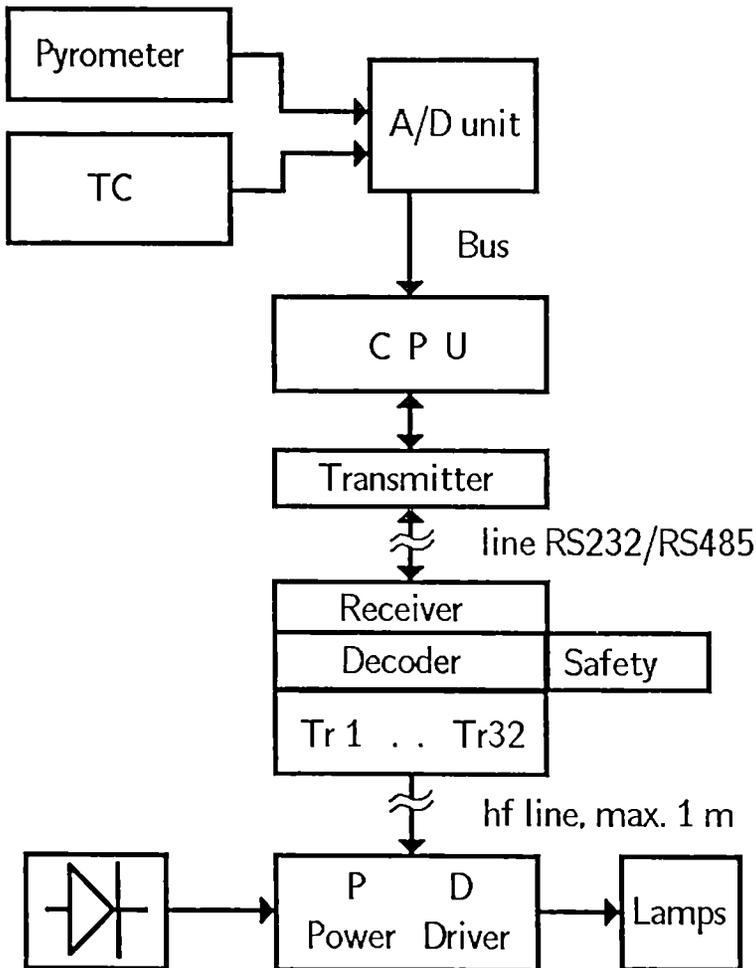


Fig. 3: Single Lamp Control - Block Circuit Diagram

Figure 3 shows the block circuit diagram for the lamp control:

An analog/digital card with up to 12 inputs digitizes the output voltage of the pyrometer and the thermocouple. Data is sent to the central processing unit (CPU) where the output data for the lamp control is calculated and sent to the transmitter (parallel interface). The transmitter transforms and coordinates the data for each lamp and sends them (serial) to the receiver. The decoded signals are controlled by a safety device. This control working with its own CPU excludes the possibility of overheating and checks and verifies the data. The signal is then sent to the lamp drivers (up to 32 lamps can be controlled separately), which provide the base current for the power drivers to control the lamp intensity by direct current (DC) rectangular pulses.

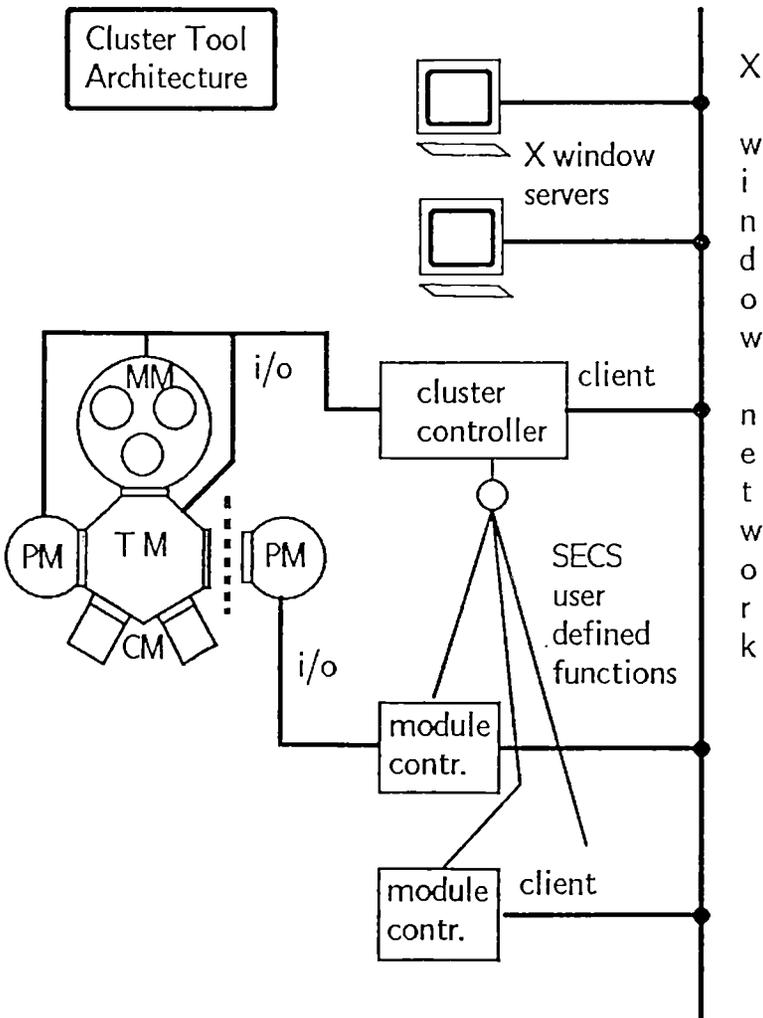


Fig. 4: Cluster Tool - Software Architecture

In contrast to lamp control by triacs DC-pulse control ensures independence from the main supply used in different countries. Noise and other disturbances can easily be filtered out: Every lamp is controlled by exact the same voltage and different AC-voltage by use of three phase power supply between the phases as well as day to day variations of the main supply are of no concern. This improves thermal homogeneity as well as reproducibility.

The communication between the RTP-module and the clusterline is managed via SECS protocol and X-windows. The final goal of this development will be a unique communication structure, i.e. the wafer processing in every process station will be controlled together with the transport system by one terminal.

Figure 4 shows a block circuit diagram of the software control. Every process module (PM) has its own module controller, communicating via SECS protocol and user defined functions with the cluster controller. The cluster controller manages the transport system, supervises the integrated processing sequences and also every individual process module controller.

X-window network enables one operator to access directly to every module controller and also to the cluster controller. This recovers costs and yields maximum output with minimal manpower.

REFERENCES:

- (1) J.R.M.Vadjikar, R.P. Roberge, T.G. Wolfe; J. Electrochem. Soc. Vol. 135, 10, p.2582
- (2) T. Nishimura, Y. Yamaguchi, H. Miytake, y. Akasera; IEEE SOS/SOI Technology Conference 1989, p. 132
- (3) Y. Ku, S.K. Lee, W. Ting, D.L. Kwong; 1989 International Symposium on VLSI Technology, System and Applications; Proceedings of Technical Papers NY, USA IEEE1989, p. 337
- (4) R. Kakoschke, E. Bußmann, H. Föll; Appl. Phys. A 50, p. 141, 1990
- (5) Technical Notice 9 by A.S.T. Elektronik GmbH
- (6) R. Kakoschke, E. Bußmann, H. Föll; Appl. Phys. A 52, p. 52, 1990
- (7) P. Vandenabeele, K. Maex; SPIE Proc., 2 - 3 October 1990, Santa Clara, p. 1393
- (8) R. Kakoschke, E. Bußmann; Mat. Res. Soc. Proc. April 1989, p. 149

COMMON LIBRARY CONCEPT AND DESIGN METHODOLOGY IN IDPS

JP Moreau
SGS-Thomson, Central R&D/DAIS
BP 217, 38019 Grenoble CEDEX
France

Project Partners:
Philips, SGS-Thomson, Siemens, Plessey, ES2
Bosch, BULL, ICL.

SUMMARY

IDPS gathers five European ASIC suppliers and three System Houses to define an advanced ASIC service, based on common concepts, libraries and design methodologies. Using the IDPS approach will give European Customers the opportunity to do portable designs, manufacturable by any of the five IDPS vendors.

IDPS design methodology is based on a top down approach, using VHDL and Synthesis tools. IDPS digital library is structured in three levels: SystemLib, MacroLib, and CoreLib and completed by a family of regular array generators. The only technology dependent parts of the Library are CoreLib and Generator Leaf cells (primitives). These parts will be optimised to each vendor's process for the best performance compromise.

This paper presents the main concepts which form the backbone of IDPS, in the field of Standards, Library Structure, Generator Concepts, Process Issues and Design Methodology.

1. Foreword

The common library concept is central to IDPS: from it, an important move forward is expected in system designer efficiency, as well as a much stronger position for European ASIC suppliers.

The library itself is the most visible part, but would be nothing if not defined in the framework of a global consistent design methodology. Thus, one of the first undertaking is to clearly identify which design methodology (ies) should be supported by IDPS.

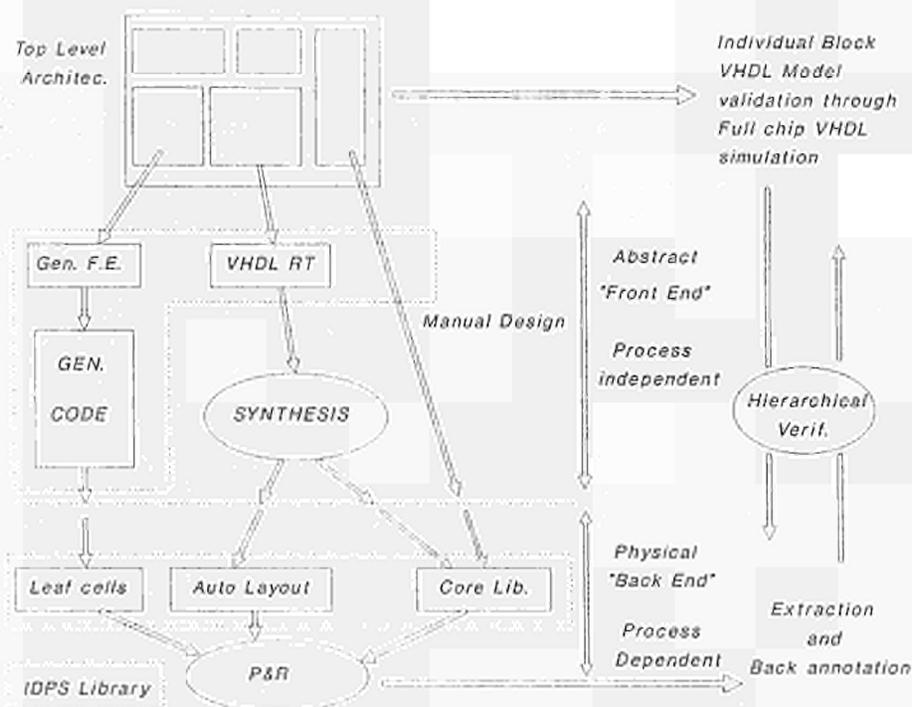
This aspect of the problem is particularly critical, since new CAD tools are now available on the market which may completely change the way designers work. Among those new tools, Synthesis and VHDL design environments are thought to play a major role for digital design. New tools are available in Analogue ("IDAC like") which may also modify the Analogue designer's life.

Considering these new opportunities, and evaluating the consequences on the IDPS strategy is of paramount importance: the development of a useful library is well known to be a huge effort, and making it compatible across several foundries even more difficult. Careful choices have to be made at the very beginning and a continuous evaluation on the consequences of new tools should be done.

2. Digital Library

2.1 Design Methodology, Basic Principles.

One of the strategic choices for IDPS is to support a consistent top-down, VHDL based design methodology. Such a methodology is based on a recursive structural decomposition into modules. Each module is defined by a VHDL behavioural model, and each step of the decomposition is validated by co-simulation against the previous step.



This, done at the beginning guarantees the validity of the chip specification, and throughout the design process, guarantees the validity of block specification. Block models may be the VHDL behavioural view of some existing library element, or the behavioural view of a block to be further partitioned or ready for synthesis (VHDL RT).

In this approach, the only technology dependent part of the library is made of the Standard cell Core library (used as a target for synthesis) and the generator's leaf cells.

The synthesis may also map on an autolayout tool. When this is possible, the target reduces to simple parameterisable transistors, allowing full mapping on any process.

The generator code is not process dependent, but is usually CAD dependent. Making generators portable is thus an important issue.

Complex predefined functions can be made available using either a synthesis path, or using some "full custom" (generator) approach. The synthesis path is preferred because of its flexibility (very cheap mapping on each vendor process, and possible exploitation of "non common" cells to better optimise the result without any decrease in generality). However, the two methods may co-exist: some complex function may be implemented as full custom by one partner, and through synthesis by another one. This

makes sense for functions already existing as background knowledge of one partner, and recognised as being of major interest by the others.

One consequence of this strategy is to reduce the number of cells (technology dependent) present in the core library. Of course, more cells may be necessary to ease the designer's task (intermediate design steps), but since the general philosophy is to make a global optimisation using synthesis tools, those cells need to be only abstract models, with possibly some timing information coming from the characterisation of a synthesized instance. As well as for complex macros, some hard implementation may exist at some vendors, but this doesn't change the general concept.

2.2 Core Library

A first kernel for the Core digital library has been defined ("IDPS CoreLib") and checked against the basic requirements of synthesis tools. This core was first chosen to match the upward compatibility criteria of each vendor Company. This guarantees the immediate portability of an IDPS design on an existing library, without even a soft mapping as initially proposed by ES2. Of course this soft mapping proposal remains for the library extension (which will be mostly "soft mapped through synthesis" in the final 0.7 micron IDPS technology also).

The upward compatibility criteria is also necessary because IDPS library is not considered by ASIC vendors as an offer independent of their normal offering or something which may be substituted to it. Rather, the IDPS library will be a subset of their total offering. Using IDPS subset will give more freedom to the customers at a cost of a small performance reduction.

IDPS CoreLib provides, in its initial definition, 18 combinatorial cells (all inverting), 8 sequential cells (latches, Flip-Flop, Scannable Flip-Flops), 2 Mux, 1 Adder, and 9 I/O Buffers, plus some ancillary functions such as Bus Hold, and Logical 0/ Logical 1 sources. Sequential cells follow a classical single clock scheme. Dual clock latches are not a part of CoreLib, but may exist as primitive objects used by macro-functions.

2.3 Non Regular Macros

For a high system design efficiency, availability of more complex functions is thought to be necessary.

Example of those functions are classical "MSI" functions (8 bit adder, 8 bit shift register, octal latches....) and "LSI" functions (serial I/O controller, parallel I/O controllers, DMA Controllers, Interrupt Controller) possibly emulating existing commercial parts. The simplest way to make a process portable version of those blocks is to define their abstract model and generate them using synthesis tools and the Core Library as the physical target. This has also the enormous advantage to provide an IDPS version, not restrained by industrial ownership problems. Of course these version will be much less optimised than the original, full custom ones. However, some experiments currently being made, prove the result to be more than acceptable for most applications.

One side effect of this approach is to allow a global optimisation: all unused functions of the macros will be filtered out by the logic optimiser, which is not possible when using fully predefined blocks.

On the other hand, the concept of abstract macros goes against the traditional "pre-characterised" philosophy: being only abstract, those macros cannot be characterised accurately on layout. This is not too much a problem for final certification of the design, since it is to be done from *physical gate net list* (after synthesis). It is a drawback

however during the design because people need to think at macros'level, also in terms of timing. The strategy is to provide for those macros default timing values calculated from their synthesized implementation, using IDPS Core Lib as a target.

The objective of IDPS is to provide first a subset of the 74xxx MSI functions, ("IDPS MacroLib"), then (during the next phase of the project), a set of LSI/VLSI functions ("IDPS SystemLib").

2.4 Generators

2.4.1 Definition

Function Generators are pieces of code generating most of the views associated with a particular function (layout, but also behavioural and timing model, possibly test model etc.). So far partners use various languages, even several for the same generator (one for behavioural model generation one for layout generation) making the transport of the code to another system highly problematic. Although the synthesis approach may sometimes compete with the generator approach, it is clear that, for highly regular structures, particularly those mapping on two dimensional arrays, a direct generation (using optimised leaf cells rather than "standard cells") will give much better results in terms of area and possibly of speed.

Porting a generator from one system to the other consist in porting its different views, or porting the code which generates those views.

From the user point, the main issue is the front-end: most generators are parameterised objects, while usual library functions are fixed (even if very complex) objects. A standard library Data-Base can store only fixed objects, or generator *instances*. The generator user interface is not a "view port" on the library data base, but the user interface of a particular CAD Tool, able to create library functions at run time. Much more sophisticated Object Oriented Data-Base would be necessary to store the generator code itself, and more sophisticated schematic capture mandatory to make access transparent to the user.

The first generation of IDPS is being built on existing CAD systems (those supported by IDPS ASIC vendors), each generator user interface will be seen as a particular CAD Tool. Attempt will be made to unify this interface, particularly when the basic resources are available from the framework, on a standardised basis.

From the Generator user interface, the user will select the parameter set he needs (including possibly technology parameters), then the generator code will generate the different views of the instance corresponding to this set of parameters. The generated views will conform to IDPS standards (and will be stored in the "work library data-base"). They will be compatible between partners, at least for all "abstract views".

From the "IDPS supplier" point of view (and for the "type 1" customers, see section 5), there is however one extra difficulty: the generator code is more than a simple (i.e. C) program. Each view shall be described using an appropriate formalism, and then this description compiled by an appropriate tool to fill in the corresponding standard views in the work library data-base. The formalism may be directly compatible with the data base format: for example, one can imagine writing a parameterised VHDL behavioural model (using "generate" construct"). It may also be specific, and this is the case for the layout views (physical and abstract).

2.4.2 Generator Physical views description formalism

The main feature of functions for which the generator approach is preferred is regularity, (if not regular, synthesis may simply be better...), thus generality will not be restrained if they are considered as simple tilers, handling "hard leaf cells". The problem limits to porting a "tiler" code, and porting - separately - the corresponding leaf cell's layout.

Functions needed to make a tiler are much more simple and in smaller number than functions needed to make a full procedural description language. Algorithmic power is also very limited. There is thus a reasonable hope that a common tiling language could be defined, and used, either as an intermediate for code translation, or in a longer term as a native language for development by most of the partners.

Following possibilities were considered:

- Defining a new -simple- language based on C (prototype: the BULL Tiler language, which is *embedded in C*);
- Using a subset of the Philips research MODGEN system (*which use a C like syntax, and a dedicated C compiler*) (system offered by Philips Research to IDPS partners), MODGEN is marketed by SSS in Dublin.
- Using a subset of STyX (*embedded in Lisp*) (system offered to IDPS partners by SGS-Thomson),
- Using an already in existence commercial language such as L from Mentor.

Solution 4 was discarded for lack of control on the standard.

From a theoretical point of view, Solutions 1 and 3 differ from solution 2 in that they are both embedded in a high level language, while MODGEN is a dedicated language, requiring a specific compiler. Consequence is that MODGEN is more secure for a "non professional programmer", as most layout designers are, at a price of possibly weak algorithmic performances. For example, if someone has in mind to develop an "intelligent" generator, including sophisticated optimisation steps, this may be simply impossible if he is restricted to the limited number of data types and algorithmic constructs available in MODGEN.

On the other hand, if the scope is to generate the layout of a pure tiler, leaving all non structural parameterisation to external tools, as it was defined at the beginning of the project, MODGEN complies with most of the criteria defined by the working group, and its C syntax makes it easily understandable by most designers.

Considering the IDPS scope, the decision was made use MODGEN as a common representation for *layout exchange* (tiling code) . Some partners consider using it *for development*, but this remains to be balanced against solutions based on more traditional commercial tools, or more powerful internal solutions.

It is essential to emphasis on the difference between a "standard representation" for layout and a layout "development environment". For example GDS-2 is a standard for describing fixed layout at polygon level. GDS-2 files may be generated from various tools using different input formalisms. Although more difficult, this may be also the case for parameterised layout: whatever will be the original formalism, *pure tilers* will be translatable (possibly manually) in MODGEN. This will not be possible for very sophisticated generators, which would have been impossible to describe in MODGEN anyway. Those "mythic" generators are not in the scope of IDPS today.

2.4.3 Generator leaf cells

Contrary to CoreLib, for which final layout is specific to each partner, geometrical standards of generator leaf cells are fixed by the generator itself, not by constraints specific to the different vendors (process design rules apart). Therefore, it makes sense to port leaf cell layout from one vendor to the other, and there is a need to describe leaf-cell layout in a process independent way. This may be achieved using symbolic layout (on a fixed grid or on a virtual grid with compaction), or using procedural layout (as it can be done in MODGEN or in STyx). There is also some possibility to use migration tools such as DESIRE marketed by DOSIS if hard layout was used. Clearly the more convenient approach is to use symbolic layout, a possibility which is offered also by MODGEN. This solution is adopted for the future, and a standard way of describing a "symbolic layout", based on ModGen will be used.

2.4.4 First Generation of Generators

IDPS Partners are developing a first generation of generators from the beginning of the project, while defining at the same time the standards to which they will have to conform. This means practically that the first generation will only conform to standard for abstract views, but not for physical views. In other words, the layout of this first generation will be hardly portable leaving open the problem of intermixability of generators coming from different vendors in the same design, and of second sourcing of design using those generators.

The problem occurs essentially at the vendor site: ASIC designers usually don't see the layout, but only the abstract "front-end" views.

The IDPS "phase 2b" solution for this problem will be to code (or to translate) the layout view in the common language to make it portable. This will be done for the preferred generators, and the choice will be made easier by experimenting with their first implementation.

The short term solution is to provide the partners with the proprietary part of the generator compiling environment, to allow each of them to generate the instances needed by a particular design. So far the Layout is not generated by the end user but by the vendor, this may be workable solution. Therefore, this intermediate solution creates a workable situation even with those proprietary layout tools implemented at only one site per partner.

2.5 The IDPS VHDL Reference Library

2.5.1 IDPS Library Standards

Standardisation is a cornerstone of IDPS. Defining standards is a prerequisite for a common Library. Library standards need to be independent from tools to guarantee a maximum freedom of tool choice between the partners and other users, and easy integration of new tools to profit from new development in CAD, while safeguarding the investment in libraries.

The IDPS Library Standards Book defines the various representation levels of a library item called "views", together with rules for the content of these views, as well as for the formats in which these views are presented.

This IDPS Library Standard makes use of internationally accepted standards such as VHDL and EDIF, to enhance the chance for international acceptance and to widen

the range of tools adapted to work with standardised library items. They also build on previous and concurrent work done in ECIP concerning the use of EDIF. EVEREST proposals for test data representation are considered, together with the proposals from the VHDL standardisation group, and from the EDIF standardisation group.

IDPS views are defined with a two level hierarchy:

– BEHAVIOUR

- High level (Petri net, algorithmic)
- Logic/Boolean
- Electrical (Timing, Power)
- Functional

– STRUCTURE

- Symbol
- Net-List

– GEOMETRY

- OutLine and Terminal
- Layout

– TEST

– SPECIFICATION/DOCUMENTATION

- Data Sheet
- QC Documentation

Choosing VHDL and EDIF is only the first step of the standardisation process. More detailed conventions are to be agreed upon. VHDL and EDIF define essentially a syntax. Defining a common semantics is the first goal of the standardisation task.

Detailed IDPS standard will be published in the IDPS Library standards Book (1)

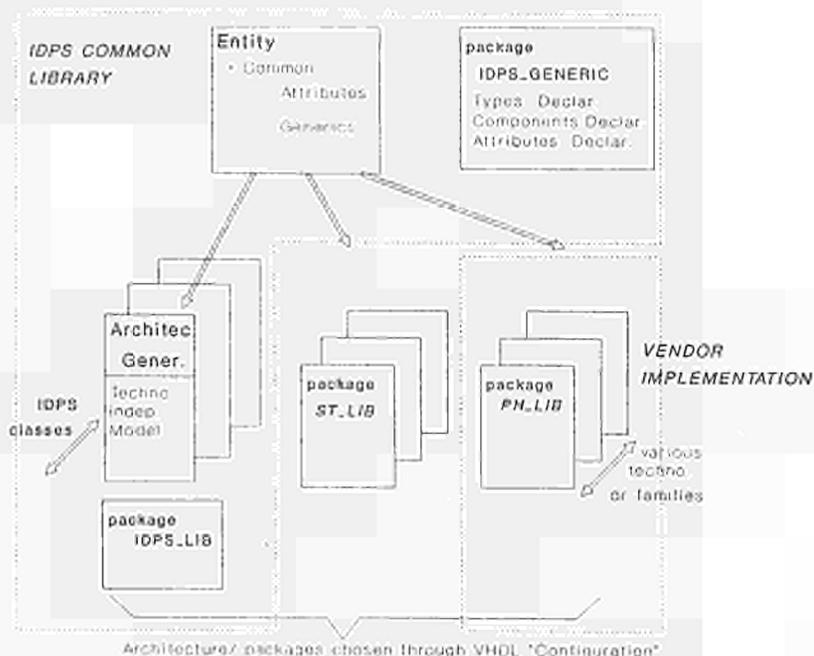
2.5.2 VHDL as formalism for storing library data

VHDL is basically a specification language. The only way to check a functional specification being simulation, VHDL is essentially considered as a language for digital modelling and simulation. However, much more than a simulation model can be described in VHDL: through the concept of "user defined attributes" almost any type of information can be introduced, to be interpreted by dedicated CAD tools (such as place and route from VHDL netlist, back -annotation from layout, generation of views for commercial systems, layout versus schematic verification, production of paper data sheet from data-base information....).

All the information needed in a data sheet may be described in VHDL language, beside the normal information required for simulation. (2)

2.5.3 The concept of VHDL reference library

The notion of VHDL configuration allow several VHDL architectures to coexist in the library description, providing support for various foundry specific or family specific data: through the "configuration" the real library (manufacturer/family) will be chosen, together with the possible redefinition of the cells and pin names (process totally invisible to the user). Each foundry will have only to customize the generic architecture to his own internal standards. This can even be done by defining foundry specific "packages", leaving a very small task of adaption.



The figure above summarises the basic aspects of the concept:

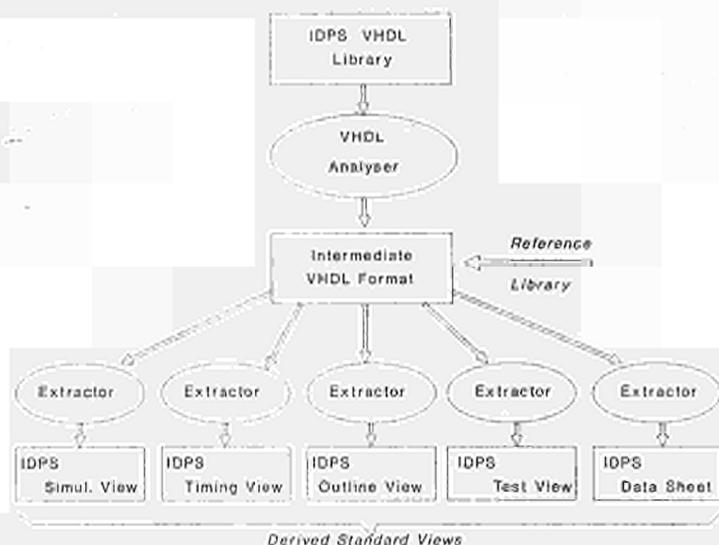
- A VHDL package "IDPS_GENERIC" declares types, attributes, and generic components,
- Default values common to all cells are defined at library level (the library itself is defined as an entity),
- Each cell entity has its own default values, and common "models" are defined in the cell's "architecture".

This information is sufficient to simulate a design, using default timing values and pre-layout default "annotation".

Vendor values will be substituted to default values using the VHDL "configuration" directive. If necessary, this could include name substitution, thanks to the VHDL "map" construct.

This gives sense to organizing a front-end CAD system around a VHDL data base. This also provides a way to implement the library in a hierarchical fashion, where a unique "extended VHDL view" is used to generate automatically the various views

defined in the IDPS Library Standards Book, giving a very clean way to manage the consistency of the library through the various releases.



This concept of "Reference VHDL Library", which may ease greatly the management of the Common Library is in discussion in between the members of the IDPS-VHDL standardisation group, and experiments are carried out using the DLS data-base (from CLSI).

Depending on the implementation of IDPS CAD systems, tool dedicated views (working representation) will be derived from the Reference Representation (VHDL Intermediate format) or from the IDPS Standard Views, derived from the reference representation.

2.5.4 Extending the IDPS Library with proprietary "macro-cells"

The minimum requirement for a cell to be usable in the context of IDPS is the availability of an IDPS behavioural view (for simulation) and an "IDPS structural view", in terms of IDPS CoreLib elements for default implementation. This latter view can be automatically generated from a RT level description by a synthesis tool. This feature can be used for two purposes:

- 1-A functionally equivalent version of a proprietary macro-cell can be made available to the IDPS partners (and users..), without disclosing physical implementation. Performances will not be as good as the original, but may be usable in many applications.
- 2-Any digital design using the full standard cell library of a particular partner may be converted to IDPS standards if an "IDPS structural view" is provided for every cell used in the design.

3. Analogue Library

Analogue design is a noticeably different task than digital design. It will not be really addressed in this paper. However, there is some evidence that "synthesis like" methodology may also be used for analogue. If this is confirmed by the work done at present

in the project, a possibility to use concepts comparable to those presented above for digital will appear.

Status of analogue libraries in IDPS is summarised below.

Two major classes of analogue libraries are planned in the project:

General purpose analogue, not requiring process option.

In this class, medium performances cells will be present. They are intended to be used in a primarily digital context, and should thus be processable in a pure digital technology.

Application oriented and general purpose cells requiring process options. These cells are more critical and will need a process with extra features (second poly, higher voltage rating...). They are not considered during the first part of phase 2 (= on going contract), partly due to the uncertainty on available processes.

It is understood that design of analogue cells will require tighter controlled process parameters, which disqualify submicron processes at present. So most design activity is taking place in 1 micron, the only way to reach usable results in the short term. Experience gained in 1μ will be used as a basis for 0.8μ as soon as good characterisation of processes are available.

However, it must be kept in mind that analogue design is much more sensitive to process parameters than digital design. Even with "compatible processes", with common layout design rules acceptable performances will only be achieved if the critical component size is optimised for each foundry set of electrical parameters, including their tolerances. Analogue cell design methodology will have to be based on the use of optimisation tools, and/or of synthesis/sizing tools such as CSEM "IDAC". This means that use of procedural layout or "pseudo automatic layout" will be required to generate as many version as available foundries.

4. Process issues

Working with the same process is not an absolute prerequisite to IDPS, the concept of abstract library and foundry mapping allows some decoupling between the "commonality of the library" and the commonality of the processes. Moreover, the way the library is going to be structured (see above) will reduce the process dependent part of the work. However, this process dependent part is still significant, and will even be more if some "handcrafted complex blocks" are going to be included in the common library. This is the reason why, in the Technical Annex, assumption was made that the various 0.7 micron processes will be "compatible", which means close enough to allow migration of a hard layout by simple sizing, or through some simple symbolic approach, without too much variation in performances (this assumption is good for digital, as said above analogue will require a more sophisticated approach).

The basis for this assumption was the concurrent work done in the Joint Logic Project, to define a common 0.8 micron process, suitable for ASIC. The first JLP target set of design rules is characterised by parameter values within a relatively broad range. This is the price to pay for commonality, while maintaining, for each participating company, a smooth transition with the previous generation of technology and products.

However, the very existence of this common set of rules guarantees that the basic assumption on process similarities is fulfilled. In other words, it will be possible to design

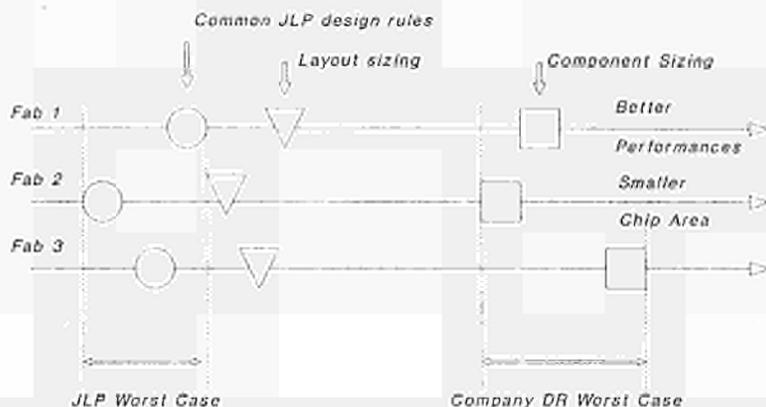
layout either with the JLP rules, or to some proprietary set and move it to JLP, or else, a JLP layout will be easily migratable to any proprietary set of rules within the partners.

This gives basically three choices:

Keeping the JLP reference design to be processed as it is.

From that reference design, automatically generate a layout conforming to the specific partner's rules.

From that reference design, used as a "topological guide", generate partly by hand (or using "smart sizing" tools), a layout with optimised component size.



The first choice is very simple, but may lead to some loss in performances in logic design (which remain to be evaluated), its usability for analog design is highly questionable.

The second should give slightly better performances and cost a little more in design effort. Since this way of doing doesn't allow to adjust critical parameters such as transistor size, it is unlikely it may improve the situation for analog design.

The third approach requires a much bigger effort, but may be the only usable for analog, and for high speed (compared to process performances) logic. It is also consistent with the "reduced library set" concept, and the exploitation of modern automatic layout generation tools.

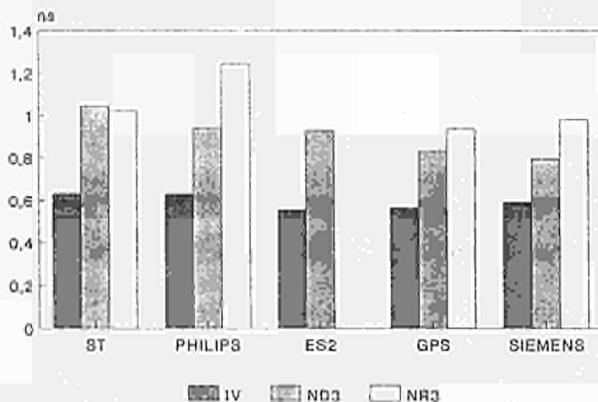
For the IDPS standard cell Core Library, the requirement is that the dimensional standards comply with the already existing choices such that the Core Library is an integral part of the total company library offer.

One major consequence of the upward compatibility (compatibility with each partner's standard cell extended offer) is the impossibility to transfer standard cell layout between partners: standard cells must comply with dimensional standards, already existing, and particular to each partner (except if some partners decide to give up with the upward compatibility criteria and to adopt another standard cell family, which, of course will not be limited to a simple "core").

For standard cells, only the third option described above is usable. For generator leaf cells, the situation is different, and the three options may be used, on a performance/area trade off basis.

This constraint of upward compatibility for standard cells may be thought to introduce severe discrepancies in performance of the same circuit produced by the various vendors. In fact this is not the case: by and large, the performance of a library is directly dependent of a kind of process "merit factor", which itself depends on the process

"generation". In other word, detailed design rules may be different, they should be as a whole consistent: if they are, process from the same generation give very comparable performances. This is demonstrated by the following figure, where performances of three basic cells of IDPS CoreLib are compared. Conditions are a three standard gate loading (to keep library compromises consistent) and a worst case voltage (4.5v) temperature (100°C) and process (slow process, estimation for a 0.7 μ process). Difference between the fastest and the slowest cell is 12.5% for Inverters and 23% for the Nand 3. Implementation is according to Partner's standards, *without any attempt to reach a common performance objective.*



5. Customer Typology and Design Methodologies

5.1.1 IDPS Baseline.

When analysing the ASIC market, three types of customers may be identified, on the basis of the design methodology they use normally:

- 1-System houses (usually big) with strong design teams owning a real semiconductor design skill: they usually design their own libraries down to transistor level.
- 2-Traditional designers (usually SMEs), just moving from Board Technology to ASIC technology, and willing to transpose their design at low investment.
- 3-System houses, not necessarily big, with no particular knowledge of silicon design, but highly skilled in modern system design techniques: They will be very pleased with synthesis tools and a pure abstract library of complex blocks.

System Houses within the IDPS Consortium (Bull, Bosch, ICL, and System Group of other Partners) are clearly from the first category, and requests issued by the first IDPS User Group meeting reflect the position of "Type 1" customers.

However, it should be emphasised this is NOT the market considered for the "IDPS Service": IDPS base line is to support the "type 3" customer, and to provide extra facilities for type 1 and 2.

For type 1:

Provide facilities to integrate a private library (conforming to basic IDPS standard requirements) in an IDPS CAD system. Provide the library itself on a form which can be integrated in a user CAD system (conforming to a minimum set of IDPS standards).

For type 2:

Provide synthesisable abstract models of popular components to map existing design, or usual design methodology on the IDPS Core Library.

5.1.2 Digital ASIC Design methodology

Base Line ..

Write a top level behavioural model in VHDL, validate by simulation in the context of use (board level). Define a set of functional test vectors, use IDPS generic types and generic components --> **This is the reference.**

define a first level partitioning (memories, processing units). Write VHDL models for those blocks, simulate the full chip, and check against the top level model using the functional test set.

For each block:

If necessary partition again hierarchically to reach a structure of blocks which are either

Functions defined in the IDPS SystemLib or MacroLib,

Functions described at RT level by the user.

make a structural model of the block using VHDL models available in the library (i), and user defined models (ii). Substitute to the behavioural model and simulate in the full chip context.

When OK, the functions described at RT level are ready to be synthesized using IDPS CoreLib as the target.

For each block to be synthesized:

Run a synthesis tool with IDPS coreLib as the target (use provided synthesis views).

Run the timing analyser on the result (or run a simulation with a timing option), then run the timing analyser at chip level, with IDPS Library default timing values.

If OK the chip is processable by any vendor in the consortium (in the limit of acceptable complexity).

-> Choose your vendor, select through the VHDL configuration control the real parameters and possibly component mapping, run a control simulation, generate an EDIF netlist: the "(Very) Quick Turn Around" starts there. The vendor will proceed with place and route, generate post-layout back-annotation, run a final control simulation with your own patterns and process samples.

If NOT OK, but marginal, there may be some possibility that one vendor can provide some slightly faster implementation of the basic library. It is a work-around, but second sourcing capability is lost.

If really out of spec, or if you want to keep multi-sourcing capability, go back to architecture, and redesign critical blocks (step C). If not enough go back to step B and use a different (more parallel) partitioning.

Remember: B and C are steps where the Architect shows his art....

Variant for type 1 customers.

Type 1 customers make their design down to layout and deal with foundries at GDS2 data level. Being responsible for the physical layout, they have a special need of "technology migratable layout" (to go easily from one vendor to the other). Most of them use a symbolic layout approach but procedural layout may also be used (see 2.4.3 above). To be consistent with the IDPS design methodology, they need their own implementation of the IDPS CoreLib. They can decide to use the implementation of one particular vendor, but this may reduce their freedom. The problem is different with "full IDPS Standard" generators. This standard makes generator layout portable (versus technology), they will have only to install the IDPS Generator standard development environment in their own CAD system (integrated or stand-alone depending on its flexibility). This is a rather ideal situation however, since the first generation of IDPS generators will not comply with the full set of standards (see 2.4.4). The situation will have to be checked on a case by case basis, but problem will be limited to porting the generator in the customer environment; the final layout being done according to each vendor technology rules, the potential problem related with the existence of a generator at only one partner's will never exist in that case.

Variant for type 3 customers.

The base line methodology, essentially top-down may be incompatible with the capability of many SME's. There is also some cases where a bottom-up approach based on standard MSI is interesting even for sophisticated people, because a board already exists to do the job, and a simple transposition makes sense.

The purpose of the IDPS MacroLib is to give to these customers an easy entry to ASIC design, using functions well known by board designers. IDPS MacroLib will be available with all the traditional data needed for schematic capture, and ported to the most popular commercial CAD systems. From those informations, a pre-layout pessimistic simulation will be possible. The resulting netlist in EDIF format will be converted by IDPS tools into VHDL, and then mapped on IDPS CoreLib and optimised using commercial (or partner proprietary) synthesis tools.

6. EXPLOITATION

IDPS vendors strategy is to support IDPS design methodology (including the common library concept as described above) as a part of their total offer. Necessary common information will be made public through the IDPS Library Book (3) where concepts and methodology will be described, and the IDPS Data Book, where common data sheets with default timing values will be given.

Front end data will be available on the most popular EWS, and user to vendor interface will be made at net list level.

First implementation of the IDPS CoreLib is already available in 1 micron technology through ES2. SGS-Thomson and GEC-Plessey Semiconductor agreed to fully second source their implementation of the IDPS Library in 0.7μ , which will be offered as a portable subset of their respective standard cell families in the first quarter of 1992. Semi second sourcing (abstract net list level) will be available from Philips and Siemens during year 92.

7. CONCLUSION

IDPS Library is much more than "Yet Another ASIC Library", it is first a unifying concept and a set of standards (in the broadest sense) allowing a system designer to design a VLSI chip down to gate netlist, without the need of choosing a particular vendor from the very beginning, and to proceed with any one of the five IDPS vendors for the following steps.

REFERENCES

- (1) IDPS Standards Book, Revision 2.0 to be published Jan 1992.
- (2) "VHDL ASIC Library Format" Steven H. Kelem, VHDL user's Workshop, Oct 1990.
- (3) IDPS Library Book, Revision 2.0 to be published Jan 1992.

INFORMATION PROCESSING SOFTWARE AND SYSTEMS

DBS3, AN IMPLEMENTATION OF THE EDS DBMS ON A SHARED-MEMORY MULTIPROCESSOR

B. BERGSTEN^a, M. COUPRIE^b, P. VALDURIEZ^c

^a Bull Research Center, Les Clayes sous Bois, France.

^b ESIEE, Noisy-le-Grand, France.

^c INRIA, Rocquencourt, France.

SUMMARY

Since 1989, the EDS project has initiated the development of a high performance, parallel database server based on a distributed-memory architecture. DBS3 is an implementation of the EDS database software on a shared-memory, commercially available multiprocessor. This paper presents the design choices, architecture and performance evaluation of the current DBS3 prototype. The major contribution of DBS3 is the compile-time optimization of both independent and pipelined parallelism, and the exploitation of shared-memory for efficient concurrent execution. The current DBS3 prototype runs on an Encore MULTIMAX multiprocessor.

1 INTRODUCTION

EDS is an ESPRIT project with BULL, ICL, Siemens and ECRC as main partners. A major focus of the EDS project is the development of a high performance, parallel database server which offers extended relational capabilities¹³. The interface language, called ESQL, is an extension of SQL with object and deductive capabilities¹⁷. The EDS database server will be optimized towards low-complexity (OLTP-like) and medium complexity (decision-support) queries because they contribute most to throughput demands. However, high-complexity (knowledge-based) queries will also be supported. It is expected to deliver up to 10 times the performance of high-end conventional mainframes of the mid-1990s. To achieve these goals, the EDS database server relies on a dedicated distributed-memory (shared-nothing) highly parallel architecture which uses large main memory.

As advocated for BUBBA⁶, GAMMA¹², DDC² and other designers, distributed-memory appears to be the only alternative for high-end systems (e.g., requiring more than 1000 TPS of the TPC-B benchmark). However, for small- to medium-end systems (e.g., < 1000 TPS), a shared-memory parallel architecture is an interesting alternative which can provide similar or better cost-performance⁴. SABRE¹⁶ and XPRS²³ are examples of shared-memory parallel database server designs. More experimental study is needed to decide which of the shared-memory or distributed-memory multiprocessor architectures is best for data management under various workloads. Intuitively, shared-memory (SM) has less extensibility and reliability because of the common memory but provides easier load balancing (critical for performance) and low-cost communication. Furthermore, it is easier to program. On the other hand, distributed-memory (DM) can provide higher scalability and reliability.

In order to get better insight in comparing the two architectures, it was decided to also prototype the EDS database software on a shared-memory, commercially available multiprocessor. We call the resulting system DBS3 (for Database System on Shared Store). Another

motivation for DBS3 was to validate various aspects of the EDS database server (essentially the ESQL compiler), which are common to both architectures. The rationale for this decision is that most of the technology designed for DM is relevant to SM as well. For instance, declustering the data among several fragments¹⁹, each accessed in shared-memory by one or more processors reduces memory access conflicts and thus increases parallelism as for DM.

The focus in DBS3 is on ESQL compilation with optimization and automatic parallelization of decision support queries, and the parallel and concurrent execution of compiled queries. To achieve its performance goals, DBS3 exploits both independent and pipelined parallelism with decentralized execution control. Overall, the major contribution of DBS3 is to show that the implementation of a distributed memory execution model on SM simplifies the compiler (essentially optimization and parallelization) and improves database management performance on SM. The optimizer extends relational query optimization to deal with object and deductive capabilities and also extends distributed query optimization to search efficiently a large space of parallel execution strategies. The parallelizer applies optimization decisions to make explicit parallelism and dataflow between operations, and optimizes the resulting program by minimizing the number of control messages. The run-time system heavily exploits the shared-memory and the advanced features of the MACH operating system [Acceta et al.,1986] to minimize the overhead of parallelism.

In this paper, we concentrate on the design choices, architecture and prototyping of DBS3 and report on our early performance experiments. Section 2 presents an overview of DBS3 design. Section 3 presents the overall architecture of the system including the ESQL compiler and run-time system. Section 4 reports on performance modeling and experiments made with the prototype as implemented on an Encore MULTIMAX 10-processor computer¹⁴. Section 5 concludes.

2 DESIGN OVERVIEW

This section gives an overview of the major design decisions which led to DBS3's architecture. Obviously, some of these decisions were also made for the EDS database server. A number of design principles were established in order to support existing standards (SQL, UNIX) and achieve high performance.

2.1 Design Principles

Since designing complex software for a parallel computer is not fully understood, it was essential to set basic design principles. They stem from the lessons learnt from previous prototypes (BUBBA, GAMMA, SABRE, DDC, XPRS and others) and the decision of not reinventing the wheel, i.e., capitalize on state-of-the-art technology in parallel DBS. Therefore, the following principles have been established.

“RISC approach”. The advantages of RISC microprocessor design are simplicity (of the basic units) and high performance (through code optimization). Following a similar approach to DBS3 simplifies as much as possible the run-time system and shifts the complexity of optimization and parallelization to the ESQL compiler.

Large main memory assumption. Given that large main memory will soon be cost-effective (wrt. disk), we assume that the greater part of the active database can be entirely cached. This simplifies both optimization and cache management. However, we also handle the case where this assumption is false but without guaranteeing optimal performance.

Rely on relational database technology. Most of the distributed and parallel database systems technology^{20,21} relies on the relational model which makes easier data fragmentation, query optimization and transaction management. With ESQL, complex values and objects are

stored in non-normalized relations so that most relational database technology can be reused.

Rely on advanced OS. For efficiency, security and portability reasons, DBS design usually redefine most OS functions for better cache and transaction management. However, the trends in UNIX-like distributed OS, e.g., CHORUS²² and MACH¹, indicate that they will provide better leverage for DBS designers (e.g., threads, large virtual memory, mapped I/Os). Thus, we strive to exploit such advanced features.

2.2 Storage Model and Execution Model

The Storage Model and the Execution Model represent the way the Compiler sees data and execution. These models are an abstraction of the actual data representation and the low-level execution. Such abstraction is mandatory to simplify the task of optimization and parallelization of queries. At this layer, concurrency control and recovery mechanisms are transparent. Thus, we consider only intra-query parallelism in this section. We chose a DM Data Model and Execution Model for SM because (1) DM provides a more constrained model thereby reducing the search space of the optimizer, (2) code fragmentation, mandatory in DM, reduces data conflicts in SM, (3) for the sake of optimization, cache coherency control in SM shares similarities with message passing in DM and (4) the DM constraints can be relaxed statically at compile-time and dynamically at run-time, thus guaranteeing optimal utilization of the SM features (fast communication through shared memory, synchronization points via shared variables and dynamic load balancing).

To achieve good load balancing and low response time in both architectures, data are horizontally partitioned in data fragments. This allows to different processors to access in parallel the same relation.

2.2.1 Storage Model

Unlike BUBBA which implemented a single-level store, we adopt a two-level store to distinguish between permanent and temporary data. Permanent data are under the control of the transaction manager and are disk resident. The ESQL user only sees permanent data. Temporary data are main memory resident (managed in virtual memory), are unshared and do not require concurrency control. Since data intensive applications require the creation of lots of transient data, specialized access methods are used for these data.

Permanent Data. The direct storage model²⁶ is chosen to implement ESQL relations because it enables efficient access to conceptual relations for the most important access patterns. With this model, a conceptual relation is stored as one direct relation and zero or more indexes. Direct relations are placed by declustering (horizontal fragmentation) based on a function (index, hash, range, ...) applied to one or more attributes. The set of fragments of a relation is called its *home*. In DM, each fragment is stored on one PE. The number of fragments and the set of corresponding PEs are functions of the size and access frequency of the relation⁸. In SM, there is no notion of physical localization for fragments. There, only the number of fragments is important. Each fragment may have zero or more indexes. These indexes are partial in the sense that they do not cover the entire relation²⁴. When there is no index, the only access method is sequential scan. One index may be primary in the sense that it optimizes data placement on disk; the other indexes are secondary.

Temporary Data. For the support of complex queries, it is often mandatory to create temporary relations and also to create temporary data structures (e.g., indexes) to optimize run-time execution. A temporary relation is fragmented as a permanent relation. To provide efficient execution of different operations on the same relation, a temporary relation can have several fragmentations, one *primary* and zero or more *secondary*. The primary home of a temporary relation corresponds to the initial declustering obtained when a relation is created. A secondary

home provides an alternative fragmentation by means of an index and, unlike in DM, without duplication. Temporary relations also support partial indexes.

2.2.2 Execution Model

The Execution Model is a parallel dataflow execution model, which supports local data operators, transfer operators and control operators. These operators are embedded in a parallel algebraic language⁷ which enables sophisticated combinations to support complex queries (e.g., multiple joins, division, fixpoint, aggregation). A program, in this language embeds its own parallel execution control using the control operators. An alternative way to coordinate the parallel execution is to rely on a distributed coordinator which is part of the system and not of the program³. This coordinator has to be general enough to deal with all situations, and therefore suffers in performance for specific cases where tuned optimizations are possible. A detailed description of how control code is added to the program, in DBS3, can be found in ⁹.

To produce a parallel program, the compiler operates in two steps. The first step produces a graph of data operators linked by arcs between these operators. The second step completes the transformation by translating the graph in the parallel language. An execution plan is initially represented as a directed graph of operators. An operator operates on fragmented relations via operator instances. A data fragment is only accessed by one operator instance at a time. This means that no specific concurrency control is needed within a parallel transaction. Each arc is labelled with the kind of synchronization there is between the operators, say Op_1 and Op_2 . There are mainly two such synchronizations: *pipe* and *seq*. *Pipe* specifies that two operators are executed in a producer-consumer mode, Op_1 producing messages to be consumed by Op_2 . *Seq* specifies that Op_2 waits until Op_1 completes before starting its own execution. Finally, each of these synchronizations may be *local* or *global*. *Local* means that the synchronization of Op_1 and Op_2 results in the independent synchronization of each couple of operator instances $\langle Op_{1i}, Op_{2i} \rangle$ and *Global* means that a global synchronization mechanism for all the operator instances is needed.

Local control is also directly added by the Compiler by combining the code of several operators together. This provides very efficient synchronization between operators, especially for pipelining. For example, the consumer code can replace directly the code producing the message in the producer code.

The control associated with a *Global Pipe* execution is ensured by means of control operators. When a producer operator instance completes, a single end-of-stream message is sent to a centralized control operator which waits for the completion of all the active producer operator instances. When this condition becomes true, the centralized control operator broadcasts an end-of-stream message to all active consumer operator instances, indicating that they will not receive any more messages. Then, when a consumer operator instance has consumed all its current messages, a single end-of-stream message is sent to a centralized control operator, and so on.

The control associated with a *Global Seq* execution is very close to that of *Global Pipe*. The difference is that the centralized control operator which detects the completion of the first operator has to broadcast a trigger message to all the second operator instances instead of an end-of-stream message. This trigger message will activate all these operators.

These global control operators are similar to the well-known *barrier* in SM architectures and they are implemented with the same efficiency. A difference is that in our implementation, all operator instances are not always active (when some data fragments are only accessed), and also several operator instances may be supported by a single thread. Our control takes into account these differences and optimizes control for each specific case.

In the following, we adopt the following notation for simplicity. We note a local synchro-

nization by a single arrow like $Op_1 \xrightarrow{pipe} Op_2$ and a global synchronization by a double arrow like $Op_1 \xrightarrow{scg} Op_2$. All the queries start and end with the following global sequential synchronization: \xrightarrow{start} and \xrightarrow{end} . If two or more operators are running locally, they are glued together by the code generator into a single code fragment. We illustrate with an underbrace the operators which are combined into a single code fragment. Let's illustrate the execution model with some examples.

Selection. If relation R is declustered in n fragments, the operation $Select(R)$ is equivalent to the union of n operations $Select(R_i)$, with $i = 1, n$, where each individual operation can be done in parallel. However, if the select predicate involves the placement attributes, fewer nodes than n (ideally one) need be involved. The associated execution graph is very simple:

$$\xrightarrow{start} \underbrace{Select\ R}_{pipe} \xrightarrow{pipe} \underbrace{Store\ Res}_{end} \xrightarrow{end}$$

If the result has to be stored where it is produced (co-located with the home of R), then the last global pipe becomes a local one, and the two last operators may be glued together in a single code fragment, as follows:

$$\xrightarrow{start} \underbrace{Select\ R\ pipe\ Store\ Res}_{end} \xrightarrow{end}$$

Join. Parallelizing binary operations is more complex since, for optimal parallelism, it requires each operand relation to be declustered the same way. For example, if R and S are both declustered in n fragments using the same function on the join attribute, the operation $Join(R, S)$ is equivalent to the union of n parallel operations $Join(R_i, S_i)$, with $i = 1, n$. We call this join an "ideal join", and its execution graph is:

$$\xrightarrow{start} \underbrace{NestedJoin\ R,\ S}_{pipe} \xrightarrow{pipe} \underbrace{Store\ Res}_{end} \xrightarrow{end}$$

All the following join strategies are based on the hash-join algorithm. The generic operator $Join(A, B)$ means: (1) redistribute (if necessary) A on the join attributes and build, on the fly, a partial index I_A per data fragment, and (2), redistribute B (if necessary) on the join attributes and in pipeline mode, probe each tuple of B using the index I_A . Note that $Join(A, B)$ does not produce the same implementation as $Join(B, A)$.

If the Compiler chooses to use such algorithm for the previous query, the produced execution graph looks will be:

$$\xrightarrow{start} \underbrace{Scan\ R\ pipe\ Build\ I_R\ scg\ Scan\ S\ pipe\ Probe\ I_R}_{pipe} \xrightarrow{pipe} \underbrace{Store\ Res}_{end} \xrightarrow{end}$$

Note that the local algorithm "NestedJoin" in the first example may be implemented as the combination of the local operators Scan, Build and Probe. If the "ideal" condition is not satisfied, parallel join algorithms^{11,25} attempt to make such a condition available by reorganizing the relations. A reorganization means dynamically creating a secondary home of a permanent relation. Since this reorganization is performed for a subsequent operation, we create on the fly a partial index per data fragment to accelerate local processing. If S is not declustered on the join attribute, but R is, we obtain what we call an "Assoc-Join". The corresponding graph is:

$$\stackrel{\text{start}}{\Rightarrow} \underbrace{\text{Scan } R \xrightarrow{\text{pipe}} \text{Build } I_R}_{\text{seg}} \xrightarrow{\text{seg}} \underbrace{\text{Scan } S \xrightarrow{\text{pipe}} \text{Probe } I_R}_{\text{pipe}} \xrightarrow{\text{pipe}} \underbrace{\text{Store } Res}_{\text{end}}.$$

If none of the relations is declustered on the join attribute, it becomes necessary to reorganize both relations, this join algorithm is called "Hash-Join". In this case, each basic operator is implemented as an independent code fragment.

2.2.3 Multi-way join strategies

In advanced database systems like DBS3, it is important to support multi-way join queries. In this section, we show that our execution model supports left-deep, right-deep, bushy query tree scheduling strategies as well as a combination of these strategies. All these strategies are based on the hash-join algorithm presented above. Hereafter, we list some execution strategies with the corresponding execution graph. We suppose here that all the relations need to be reorganized, so all synchronizations are global. We also omit the start and end synchronizations.

Left-deep strategy. The execution graph for " $Join(Join(Join(A, B), C), D)$ " is:

$$\begin{aligned} & \text{Scan } A \xrightarrow{\text{pipe}} \text{Build } I_A \\ \dots & \xrightarrow{\text{seg}} \text{Scan } B \xrightarrow{\text{pipe}} \text{Probe } I_A \xrightarrow{\text{pipe}} \text{Build } I_{T1} \\ \dots & \xrightarrow{\text{seg}} \text{Scan } C \xrightarrow{\text{pipe}} \text{Probe } I_{T1} \xrightarrow{\text{pipe}} \text{Build } I_{T2} \\ \dots & \xrightarrow{\text{seg}} \text{Scan } D \xrightarrow{\text{pipe}} \text{Probe } I_{T2} \xrightarrow{\text{pipe}} \text{Store } Res \end{aligned}$$

Right-deep strategy. The execution graph for " $Join(A, Join(B, Join(C, D)))$ " is:

$$\left. \begin{array}{l} \text{Scan } A \xrightarrow{\text{pipe}} \text{Build } I_A \\ \text{Scan } B \xrightarrow{\text{pipe}} \text{Build } I_B \\ \text{Scan } C \xrightarrow{\text{pipe}} \text{Build } I_C \end{array} \right\} \xrightarrow{\text{seg}} \text{Scan } D \xrightarrow{\text{pipe}} \text{Probe } I_C \xrightarrow{\text{pipe}} \text{Probe } I_B \xrightarrow{\text{pipe}} \text{Probe } I_A \xrightarrow{\text{pipe}} \text{Store } Res$$

In the real world, some relations are distributed on the join attribute, and some not. To take advantage of these characteristics, it is sometimes better to adopt partially one strategy, and partially another. Suppose the compiler selects the strategy: " $Join(A, Join(Join(B, C), D))$ ", by building indexes on the smaller relations. Here B is smaller than C , $Join(B, C)$ is estimated to be smaller than D , etc. The corresponding graph is the following:

$$\left. \begin{array}{l} \underbrace{\text{Scan } A \xrightarrow{\text{pipe}} \text{Build } I_A}_{\text{pipe}} \\ \underbrace{\text{Scan } B \xrightarrow{\text{pipe}} \text{Build } I_B \xrightarrow{\text{seg}} \text{Scan } C \xrightarrow{\text{pipe}} \text{Probe } I_B \xrightarrow{\text{pipe}} \text{Build } I_T}_{\text{seg}} \end{array} \right\} \xrightarrow{\text{seg}} \underbrace{\text{Scan } D \xrightarrow{\text{pipe}} \text{Probe } I_T}_{\text{pipe}} \xrightarrow{\text{pipe}} \dots \xrightarrow{\text{pipe}} \underbrace{\text{Probe } I_A \xrightarrow{\text{pipe}} \text{Store } Res}_{\text{pipe}}$$

This query has been executed in our prototype, more details can be found in section 4.

3 DBS3 ARCHITECTURE

DBS3 is based on a Client/Server organization. The users are viewed as processes, possibly running on workstations, interacting with the database server processes, running on the parallel

machine. DBS3 is composed of three sub-systems, the Session Manager, the Request Manager and the Data Manager (see Figure 1) which rely on MACH operating system. The Session Manager is in charge of connecting and disconnecting each new user with a Request Manager module. The Request Manager receives the user requests and processes them using the Data Manager services. To improve run-time execution, queries are translated by the Request Manager in compiled C code. This code is dynamically linked with the *Execution Support* which contains several libraries (e.g., access methods and synchronization functions), and executed inside a MACH task, called an *Execution Task*. An Execution Task is directly connected with the user application to allow direct communication of results. It relies on the Page Manager which provides concurrency control and recovery. The Data Manager sub-system includes the Execution support and the Page Manager.

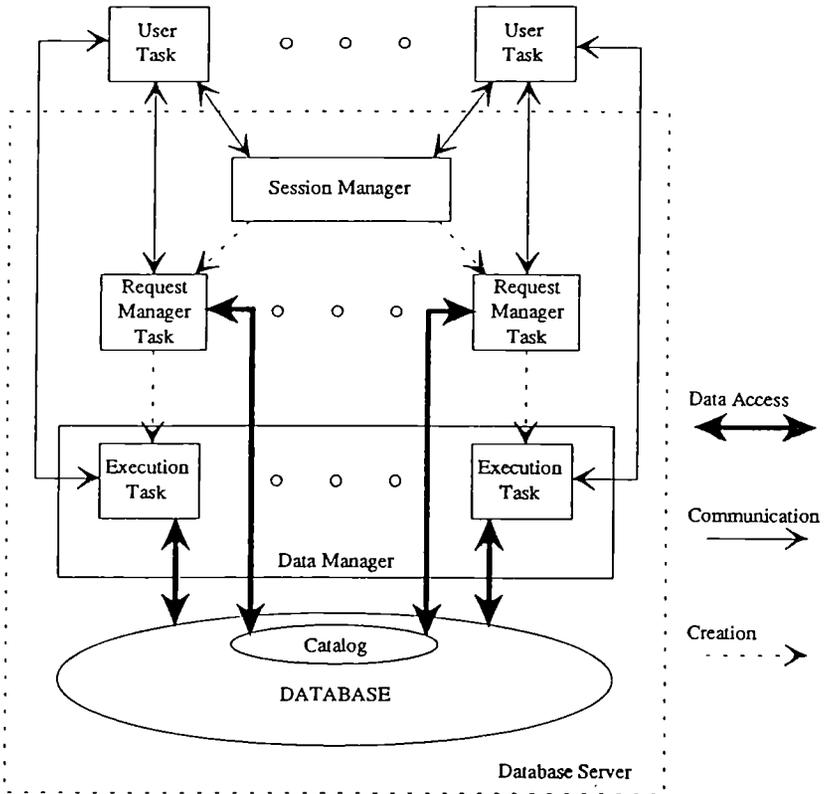


Figure 1: General Architecture of DBS3

3.1 Request Manager

The Request Manager provides the entry points to DBS3 for the application. It receives requests from the application to perform various tasks related to ESQL compilation and execution. Depending on the request (e.g., prepare-query, run-query), it activates the various compilation phases, triggers execution in the DM and returns the results as well as error codes to the application. The Request Manager supervises query execution and triggers the recovery procedure when a query fails.

The ESQL compiler transforms a DML query into an optimized object module. In the case of a DDL query request, the query is simply interpreted by the Request Manager which directly calls the catalog manager in order to update the catalog. The compilation process is divided between semantic analysis, rewriting, optimization, parallelization, and code generation.

The Rewriter rewrites the algebraic program expressed on conceptual data into another equivalent one which is simpler and logically optimized. The Rewriter implements a rule-based approach to gain extensibility¹⁵. The Optimizer takes the final optimization decisions for executing the algebraic program in the Data Manager and integrates them in an annotated algebraic program expressed on physical data. These decisions concern primarily the join ordering, the selection of the best access method to access a relation, and the choice of the best algorithm per operation. The Optimizer follows an object-oriented design approach to gain extensibility while satisfying various efficiency requirements¹⁸. The Parallelizer translates a completely optimized algebraic program into a parallel program according to the Execution Model described in section 2.2.2. The Code Generator generates an executable object module for the Data Manager. This involves producing a number of object code fragments, each one corresponding to one or more parallel operations. This code module can be compiled and linked in the usual way.

3.2 Data Manager

The Data Manager provides all the functions of the database system needed to support the execution of several parallel queries. Each query, viewed as a compiled program, is included in a transaction. Queries of the same transaction are explicitly serialized by the user application. The Data Manager performs transaction management and provides also an execution environment for the execution of parallel compiled queries.

The compiled code is dynamically linked with a set of library functions, grouped in the Execution Support component. Some of these functions rely directly on MACH, and some on the *Page Manager*.

The “Execution Support” functions simplify the work of the compiler (especially the code generator), and hide low-level complexity. To reduce globally the size of all the concurrent tasks, we factorize as much as possible the common functions and keep them in libraries shared by all Execution Tasks.

The Execution Support Libraries implement permanent access methods, temporary access methods, run-time functions, communication functions and control functions.

The Page Manager provides direct support for access methods to permanent and temporary data. It uses standard techniques (e.g., dynamic 2PL, two step-commit) as well as specific techniques for main memory data management.

4 PERFORMANCE EXPERIMENTS

In this section, we report on early performance experiments with the DBS3 prototype. We concentrate on the performance of the execution layer, which is a good basis to predict the overall system performance, and also provides feed-back for better optimization and performance tuning.

The target machine for the testbed is the multiprocessor Encore MULTIMAX 520. This machine is configured with 10 NS32532 processors (8.5 Mips, each having 256 KB cache memory), 96 MB main memory and 1 GB disk storage. Processors, memory and I/O boards are interconnected by a 100 MB/s bus.

In order to get insight in the prototype's behaviour, we experimented with three kinds of operations in increasing order of complexity: the basic operators (scanning a relation, building an index, etc), a single join operation, and a more complex query with three joins. Since the focus

in DBS3 is on decision-support queries, the cost metrics used in the performance experiments is response time.

For non-basic operations, we give approximate response time formulas, based on the critical parameters (such as relation sizes and number of available processors), and on constant times (measure of basic operator costs). To keep these formulas simple, we discard “second-order” terms.

The experimental database was generated automatically, following the specifications of the standard Wisconsin Benchmark⁵. Tuples are 208 bytes long, and relations are 10,000 and 100,000 tuples. The results of queries are stored in relations but not printed. Relations are declustered by hashing on a key attribute (the first and second attributes of the Wisconsin Database tuples may be used as keys). The number of buckets is always 60 in these experiments, because this number of buckets provides a good load balancing with either 2, 3, 4, 5, 6, 10, 12, 15, 20 or 30 threads. We repeated all the experiments with 12 buckets (divisible by 2, 3, 4, 6), without notable changes in the results.

As a consequence of the large main memory assumption, all the experiments described here involve relations that are already cached. Thus, no disk I/O is taken into account. The parallelization of our operations relies on the MACH thread management system. An important assumption used in deriving cost formulas is that MACH effectively distributes the work between the available processors, when several threads are programmed to work concurrently. We finally compare estimated results (obtained with the cost formulas) with real measured response times. We show that the measured speedup is almost as good as the estimated one.

4.1 Basic operators

The most important basic operators are: scan, build, probe, pipe and control operators. These operators can be combined together to build parallel database programs.

The *scan* operator is used to access sequentially each tuple of a relation (or relation part). Its time consists of the initialization time and the time to access the next tuple. Except for small relations, the initialization time is negligible. The *build* operator builds an index in main memory, given a set of tuple references and a key attribute. In the current version of the prototype, only balanced binary tree indexes are implemented. As for the scan operator, we will neglect the cost of initializing the index structure. The *probe* operator is used to search a key value in an index structure.

The *pipe* operator is used for communicating intermediate results between two threads. Note that in SM, only pointers need to be passed, not full tuples. The work done by this operator includes the management of circular FIFO buffers, and the use of shared resources protected by mutual exclusion locks (spinlocks).

Control operators are generated statically by the parallelizer. Their role is to start, synchronize and terminate the different threads involved in a parallel program. These operators are efficiently implemented in DBS3 by using shared variables and MACH signals. Because their cost does not depend on the number of processed tuples, we will not take control cost into account when dealing with large relations.

To these basic times, we must add the time to initialize a new thread, which is not negligible in regard to our execution times, and the time to add a new item to the result of an operation. Table 1 gives the times measured for these basic operators.

T_s	Time to get next tuple in a scan operation	$11\mu s$
T_b	Time to build a N reference index	$K_b.N.\log_2 N$
K_b	Constant for T_b formula	$6.5\mu s$
T_p	Time to probe a tuple in a N reference index	$K_p.\log_2 N$
K_p	Constant for T_p formula	$3.7\mu s$
T_c	Time to communicate a tuple reference in pipe	$50\mu s$
T_i	Time to initialize a new thread	$12000\mu s$
T_a	Time to add a new item to the result	$40\mu s$

Table 1: Basic Operator Times

For the build and probe operator, a straightforward complexity analysis of the algorithms gives equations of the form: $T_b = K_b.N.\log_2 N$ and $T_p = K_p.\log_2 N$, and we identified the values for K_p and K_b by the following set of measurements:

# of tuples	10^3	10^4	10^5	10^6
K_b (measured)	6.58	6.66	6.45	6.53
K_p (measured)	3.35	3.70	3.78	3.77

In the following, we will note $\|R\|$ the number of tuples in relation R . We will note N_p the number of available processors. In our testbed, $N_p = 10$.

4.2 Pipelined operators

Let us begin with a simple case, involving only two operators joined by a pipeline. The first operator is a scan, and the second one is a build. Let us assume that the relation to be scanned is not hashed on the key attribute that serves to the index construction. Thus, the pipeline between the two operators redistributes the tuple references found by the scanning threads to the threads that are building partial indexes. It also protects index structures from being updated simultaneously by several “building” threads. The following execution graph illustrates this operation:

$$\stackrel{\text{start}}{\Rightarrow} \underbrace{\text{Scan } R}_{\text{pipe}} \stackrel{\text{pipe}}{\Rightarrow} \underbrace{\text{Build } I_R}_{\text{end}}$$

Let $N/2$ be the number of threads executing the scan operator. Let us assume that the same number $N/2$ of threads execute the build operator. As N threads need to be created, a first component of the estimated response time is:

$$N.T_i$$

The total time spent to scan relation R is $\|R\| . T_s$. The total time consumed by the build operator is $K_b \|R\| . \log_2 \|R\|$. Since N threads are used to execute these operators, but only N_p physical processors are available to share the work, we may estimate the second component of the estimated response time:

$$\frac{T_s \|R\| + K_b \|R\| . \log_2 \|R\|}{\min(N_p, N)}$$

Finally, we must add the time for pipelining which is globally: $T_c \|R\|$. This time might be divided by the number $N/2$ of operator couples (scan—build), but there cannot be more than $N_p/2$ such couples executing on separate physical processors. Thus, the last component is:

$$\frac{T_c \|R\|}{\min(N_p/2, N/2)}$$

To summarize, the estimated response time for this operation is given by the formula:

$$T = N.T_t + \frac{\| R \| (T_s + K_b \log_2 \| R \|)}{\min(N_p, N)} + \frac{T_c \| R \|}{\min(N_p/2, N/2)}$$

We can now compare this estimation to the response time actually measured. Figure 2 shows the curves obtained from these formulas and measures, with $\| R \| = 100,000$.

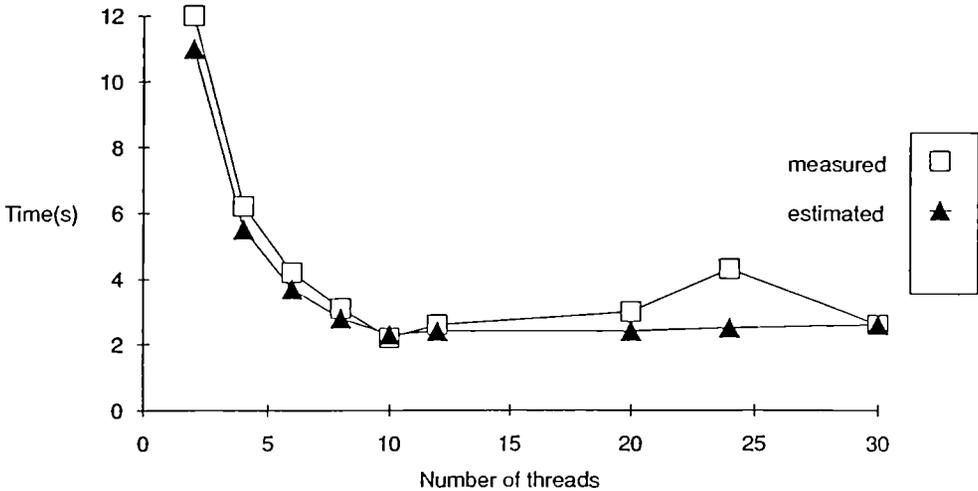


Figure 2: scan/pipe/build—10 processors

Obviously, the estimation is optimistic, since we divide the total working time by the number of processors doing the work, thereby assuming perfect parallelism. Nevertheless, the relative error ratio is only 10 % up to ten couples of threads, and the speedup for $N = 10$ (with 10 processors) is 6.7 .

It is interesting to compare the performance of two pipelined operators with the performance of one single operator doing the same work. Since write conflicts must be avoided while building the partial indexes, a single operator must lock the partial index during the update. With the pipeline, each building thread "owns" a number of partial indexes, so that no update locks are needed. The only shared resources are the FIFO queues, and locking time for a message arrival is much shorter than locking time for an index update. Thus, one could expect that the overhead of queuing messages is compensated, for this kind of operation, by the diminution of locking time. Figure 3 shows that with 8 and more threads, two pipelined operators give better response time than one composed operator.

4.3 Join Query

The join operation, because of its strategic importance in relational systems is a good indicator of relational database system performance.

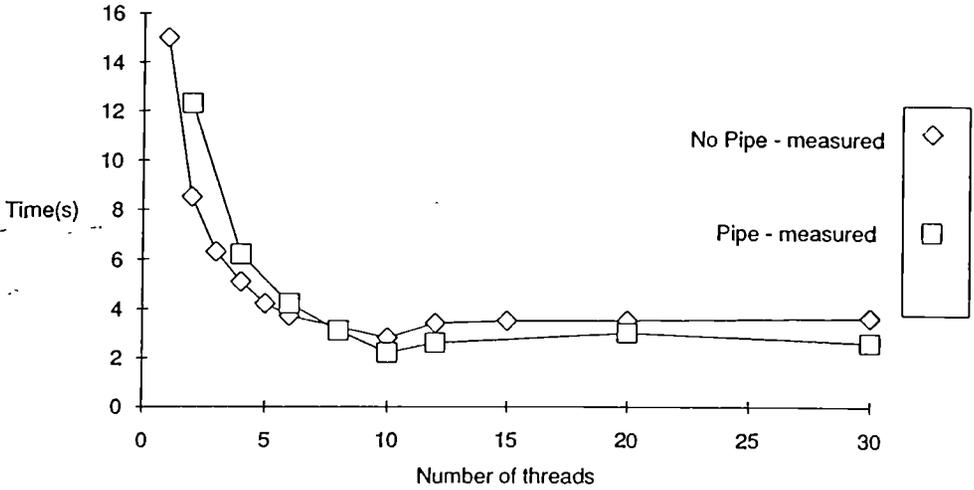


Figure 3: scan/build with and without pipe—10 processors

In the following, we study the join of a 100,000 tuple relation (let us call it A) with a 10,000 tuple relation (let us call it B). The result R is a 10,000 tuple relation. This corresponds to the “JoinABprime” operation in the Wisconsin Benchmark. In a first step, B is scanned and a temporary index I_B is built on the join attribute. In fact, I_B is composed of 60 partial indexes, and each of these partial indexes can be built independently. In a second step, A is scanned and every tuple of A is sent by pipe to be probed against the convenient part of I_B . To know what partial index to use, we simply apply the hash function to the join attribute. Then, matching tuples are stored in a temporary relation. Between these two steps, a synchronization point is introduced, because the index must be completed before the probe operation begins. The execution graph for this operation is:

$$\overset{start}{\Rightarrow} \underbrace{Scan\ B \xrightarrow{pipe} Build\ I_B}_{seq} \xrightarrow{seq} \underbrace{Scan\ A \xrightarrow{pipe} Probe\ I_B \xrightarrow{pipe} Store\ R}_{end}$$

We need to allocate two sets of threads to execute such operation: a first set of N threads will execute the first step, and a second set of N threads will be shared to execute the $scan\ A$ and the $probe\ I_B$ — $store\ R$ operators. As for the previous operation, we derive the following response time estimation:

$$T = 2N.T_i + \frac{\|B\| (T_s + K_b \log_2 \|B\|)}{\min(N_p, N)} + \frac{\|A\| (T_s + K_p \log_2 \|B\|)}{\min(N_p, N)} + \frac{\|A\| T_c}{\min(N_p/2, N/2)} + \frac{\|R\| T_o}{\min(N_p, N/2)}$$

Figure 4 compares the estimated times with the measured ones. We obtain a speedup of 7.8 with 10 processors, for $N = 10$ (calculated between $N = 2$ and $N = 10$). The best measured time for this JoinABprime is 1.8 seconds.

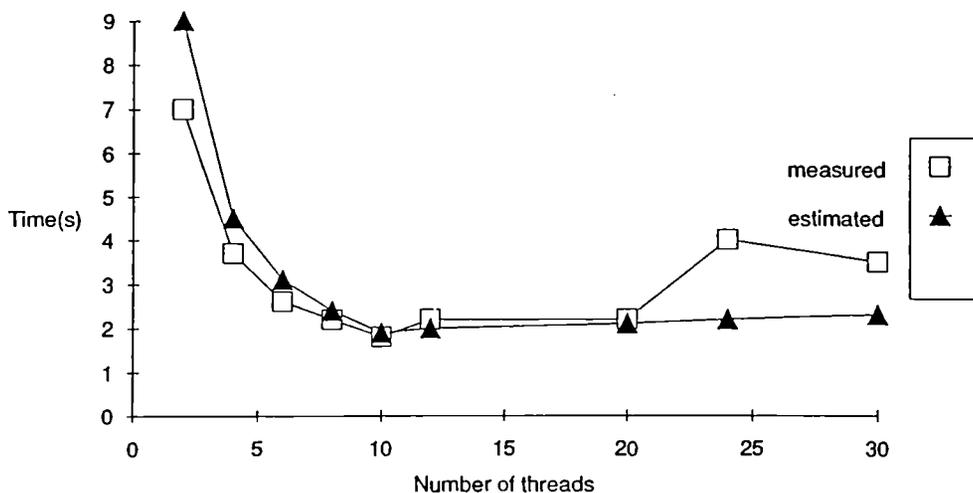


Figure 4: join query—10 processors

4.4 Complex query

In complex queries involving several operators, our execution model allows to pipeline intermediate results, saving both temporary storage cost and synchronization cost. Another advantage of pipeline versus intermediate result storage is that the time to produce the first piece of result is usually much shorter.

In this last experiment, we programmed the query example presented in Section 2:

$$R = \text{join}(A, \text{join}(\text{join}(B, C), D))$$

with $\|A\| = 10,000$, $\|B\| = 10,000$, $\|C\| = 100,000$, $\|D\| = 100,000$ and $\|R\| = 10,000$.

As for the previous operations, the response time estimation was derived in a similar way. These estimations and measures are compared in Figure 5. Note that the Number of threads reported in the Figure corresponds to the number of simultaneously active threads.

4.5 Discussion

These experiments show that data declustering in SM provides a simple way to parallelize database programs. They also show that pipelining incurs some overheads, but it also has benefits that have not been completely explored. We did show that pipelining between operators can reduce the locking time in case of write conflicts. We believe that pipelining is useful to associate program code close to the data fragments, thereby providing better locality of data references in the processor caches. This is an important factor since most SM machines are designed to support cache miss rates which are lower than 5%. Future experiments with 20 processors are planned to confirm this idea.

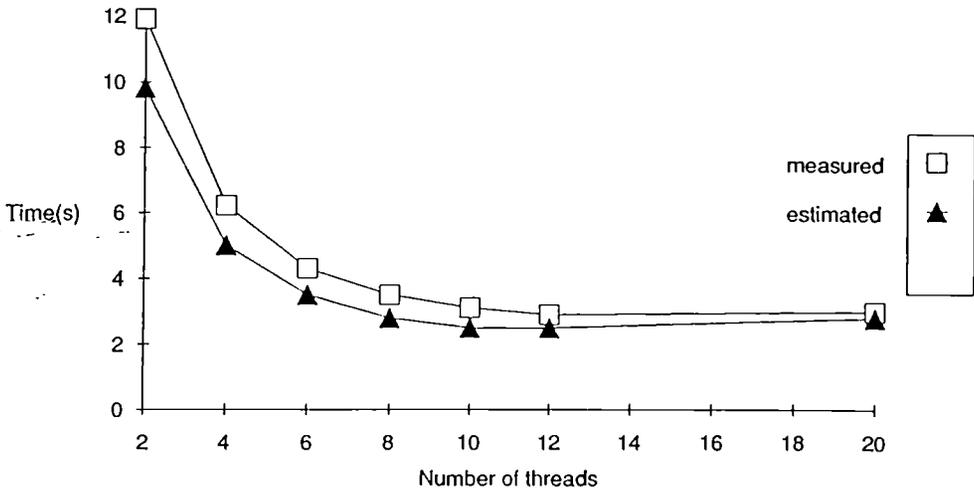


Figure 5: complex query—10 processors

5 CONCLUSION

DBS3 is a shared-memory, parallel database system whose goal is to provide high-performance for small- to medium-end systems (e.g., requiring less than 1000 TPS). DBS3 supports ESQL, a conservative extension of SQL with object-oriented and deductive database capabilities. It is optimized towards both OLTP and decision-support queries and can also support more complex knowledge-based queries.

In this paper, we have presented the design decisions, execution model and architecture of DBS3, and reported on early performance experiments with the current prototype. We focused on decision-support queries. We consider these to be the most significant contributions of the DBS3 project:

- **Parallel execution model.** DBS3 implements a parallel dataflow execution model based on a declustered storage model which provides both independent and pipelined parallelism with decentralized execution control. Compared to other shared-memory database systems such as SABRE and XPRS, this leads to much higher intra-operation parallelism.
- **Parallelizing compilation.** The ESQL compiler supports several optimization strategies to search efficiently a large space of parallel execution plans, and generates a low-level, optimized parallel program which minimizes also the number of control messages.
- **Run-time system.** The run-time system exploits much the shared-memory and the advanced features of the MACH operating system to minimize the overhead of parallelism. Furthermore, it implements a two-level store to optimize access to temporary relations in main memory.

The early implementation of the DBS3 prototype together with a performance modeling and measurement activity was useful to show two important results. First, the implementation of a parallel execution model with declustering (initially proposed for DM) does simplify the

compiler and does improve database management performance by allowing intra-operation parallelism. Second, extensible design of the compiler where implementation-dependent knowledge is abstracted (essentially in the optimizer and code generator) is a good idea. Two versions of the compiler differing only in the cost model and code generator have been produced, one for the DBS3 prototype and one for the EDS (distributed memory) database server.

Our initial performance experiments were done to study response time for single-user queries. They show a good intra-query parallelism. They also partly confirm our intuition about the performance gains that can be obtained from pipelining, and future experiments are planned to study the effect of pipelining on data locality in caches.

Acknowledgements

The authors wish to thank Jean-Marie Nicolas for his support in the DBS3 activity at BULL. They also want to thank G. Gardarin, M. Lopez and all the other members of the EDS project. In particular, the ICL team contributed to the design of the execution model and other important aspects. The DBS3 run-time system was implemented at BULL by L. Bouganim, P. Casadessus, B. Dageville and D. Terral.

References

- [1] M. Accetta et al., "MACH: a New Kernel Foundation for UNIX Development", Summer USENIX Conf., July 1986.
- [2] B. Bergsten, R. Gonzalez-Rubio, B. Kerhervé, J. Rohmer, "An Advanced Database Accelerator", IEEE Micro, October 1988.
- [3] B. Bergsten, M. Couprie, R. Gonzales-Rubio, B. Kerherve, M. Ziane, "Language Levels and Computational Model for a Parallel Database Accelerator", Int. Workshop on Database Machines, Deauville, France, June 1989.
- [4] A. Bhide, M. Stonebraker, "A Performance Comparison of Two Architectures for Fast Transaction Processing", IEEE Int. Conf. on Data Engineering, Los Angeles, February 1988.
- [5] D. Bitton, D.J. DeWitt, C. Turbyfill, "Benchmarking database systems - a systematic approach", Int. Conf. on VLDB, Florence, Italy, October 1983.
- [6] H. Boral, W. Alexander, L. Clay, G. Copeland, S. Danforth, M. Franklin, B. Hart, M. Smith, P. Valduriez, "Prototyping Bubba, a Highly Parallel Database System", IEEE TKDE, Vol. 2, No. 1, 1990.
- [7] C. Chachaty, P. Borla-Salamet, B. Bergsten, "Parallèle LERA: un Langage pour l'Execution Parallèle de Requêtes Relationnelles", BD3 Database Conf., Montpellier, France, September 1990.
- [8] G. Copeland, W. Alexander, E. Boughter, T. Keller, "Data Placement in Bubba", ACM SIGMOD Int. Conf., Chicago, May 1988.
- [9] B. Dageville, C. Chachaty, P. Borla-Salamet, "Compiling Control into Queries for Parallel Execution Management", BD3 Database Conf., Lyon, France, September 1991.
- [10] S. Danforth, P. Valduriez, "A Fad for Data-Intensive Applications", To appear in IEEE TKDE, 1991.

- [11] D.J. DeWitt, R. Gerber, "Multiprocessor Hash-Based Join Algorithms", Int. Conf. on VLDB, Stockholm, August 1985.
- [12] D.J. DeWitt et al., "The Gamma Database Machine Project" IEEE TKDE, Vol. 2, No. 1, 1990.
- [13] EDS Database Group, "EDS - Collaborating for a High-Performance Parallel Relational Database", ESPRIT Conf., Brussels, November 1990.
- [14] "Multimax Technical Summary", ENCORE Computer Corporation, 1989.
- [15] B. Finance, G. Gardarin, "An Extensible Query Rewriter with Object and Deductive Capabilities", IEEE Int. Conf. on Data Engineering, Kobe, Japan, April 1991.
- [16] G. Gardarin, P. Bernadat, P. Valduriez, Y. Viemont, "Design of a Multiprocessor Database System", IFIP World Congress, Paris, June 1983.
- [17] G. Gardarin, P. Valduriez, "ESQL : an Extended SQL with Object and Deductive Capabilities", Int. Conf. on Database and Expert System Applications, Vienna, Austria, August 1990.
- [18] R. Lanzelotte, P. Valduriez, "Extending the Search Strategy in a Query Optimizer", Int. Conf. on VLDB, Barcelona, Spain, September 1991.
- [19] M. Livny, S. Khoshafian, H. Boral, "Multi-disk Management", ACM SIGMETRICS Conf., Banff, Alberta, 1987.
- [20] T. Özsu, P. Valduriez, "Principles of Distributed Database Systems", Prentice Hall, 1991.
- [21] T. Özsu, P. Valduriez, "Distributed Database Systems: Where Are We Now?", To appear in IEEE Computer, 1991.
- [22] M. Rozier et al., "CHORUS Distributed Operating Systems", Computing Systems, Vol. 1, No. 4, 1988.
- [23] M. Stonebraker et al., "The Design of XPRS", Int. Conf. on VLDB, Los Angeles, September 1988.
- [24] M. Stonebraker, "The Case for Partial Indexes", Report ERL M88/62, University of California, Berkeley, June 1988.
- [25] P. Valduriez, G. Gardarin, "Join and Semijoin Algorithms for a Multiprocessor Database Machine", ACM TODS, Vol. 9, No. 1, March 1984.
- [26] P. Valduriez, S. Khoshafian, G. Copeland, "Implementation Techniques of Complex Objects", Int. Conf. on VLDB, Kyoto, August 1986.

AN INTRODUCTION TO DISTRIBUTED PROGRAMMING IN REX

Jeff KRAMER, Jeff MAGEE, Morris SLOMAN, Naranker DULAY,
SC. CHEUNG, Stephen CRANE, and Kevin TWIDLE
Department of Computing,
Imperial College of Science, Technology and Medicine,
180 Queen's Gate, London SW7 2BZ, UK.
(jk@doc.ic.ac.uk)

ABSTRACT

Configuration Programming is an object-based approach to distributed programming. The main principle underlying this approach is that programs should be designed, constructed and modified as a structural configuration of interconnected component instances. Program structure is described by a separate explicit configuration language, while the components themselves may be programmed in a range of heterogeneous programming languages. This approach is central to the ESPRIT II project, REX, on reconfigurable and extensible parallel and distributed systems. This paper provides an overview of the main concepts underlying REX, illustrating their use in the Darwin configuration language for describing overall system structure and dynamic configuration. The approach is justified and briefly compared to object-oriented programming.

1. INTRODUCTION

The REX²⁵ project, "Reconfigurable and Extensible Parallel and Distributed Systems", aims to develop an integrated methodology and associated support tools for all phases in the development and management of parallel and distributed systems. Ten European partners from industry and university research groups are involved¹. The project started in May 1989 and is to run for a total of 5 years. Two large demonstrators in the telecommunications and Computer Integrated Manufacturing (CIM) areas act as a focus for the work and as a means for demonstrating the techniques and tools developed.

The key feature of the REX approach is its emphasis on **configuration structure**. Practical experience in software engineering has taught us that complex systems can be built and managed provided we adhere to some sound principles. Software modularity is essential, and permits the use of composition to form a system. In particular, distributed systems are conveniently described, constructed and managed in terms of their software structure. Descriptions of the constituent software components and their interconnection patterns provide a clear and concise level at which to specify and design systems, and can be used directly by construction tools to generate the system itself. In many cases - particularly embedded applications - it is the structure of the application itself which is used to dictate the structure of the resultant system. This approach is referred to as "configuration programming"^{11,13}. Extension and

¹ The REX partners are Imperial College and PRG Oxford in the UK; GMD Karlsruhe, Siemens, Stollman, Karlsruhe University, T.U.Berlin and 2i in Germany, Intracom (and Intrasoft) in Greece, and Tecsi in France. Dirk Dettmann of Stollman is the Project Manager, and Jeff Kramer of Imperial College is the Technical Director.

evolution of the system can be achieved by making changes to the system configuration¹⁰.

The premise of the approach is that a separate, explicit structural (configuration) description is essential for all phases in the software development process, from system specification as a configuration of component specifications, to evolution as changes to a system configuration. Our development process is essentially "constructive"¹², emphasising the satisfaction of system requirements by composition of components. The formal view of this constructive approach is that of system verification by showing satisfaction of the system specification as a composition of component specifications. This approach thus emphasises **composition** as a basic operation of configuration, with the need to support composition of actual system components and specifications.

Hence, system configuration (structure) is recognised as the unifying framework in REX. As stated in the project synopsis, "...the notion of the system as a configuration of modular software and hardware components will be used as the framework on which to hang the research work on system specification, analysis, construction and modification"²⁵. We believe that this emphasis on configuration structure leads to clear and flexible designs, and results in distributed object-based systems which are comprehensible and maintainable. The structural view is particularly useful in facilitating change and evolution in the form of dynamic configuration. Furthermore, REX components may be programmed in a range of heterogeneous programming languages.

Overview of the Rex Software Architecture

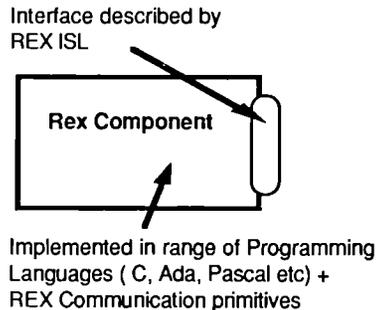


Figure 1 - Program Components

Central to the REX project is the view of distributed and parallel systems as interconnected sets of component instances. Program components are objects² which encapsulate state and have well defined interfaces specified by an Interface Specification Language (ISL). The functionality of a program component may be implemented in a range of programming languages; however, interaction with other components utilises a common set of communication primitives supplied by the REX runtime system (Figure 1). These communication primitives provide for both synchronous (bidirectional) and asynchronous (unidirectional) communication. Components in REX are types (c.f. classes) from which more than one component instances can be created.

² Wegner²⁷ makes a distinction between object based, class based and object oriented languages. Object based languages support encapsulation in the form of objects, class based extends this to include classes and object oriented further extends this to include inheritance. Strictly speaking, our language can be classified as class based.

Component instances execute in parallel. The sets of instances which exist in a system together with their interconnections (Figure 2) are specified in a separate configuration language - Darwin.

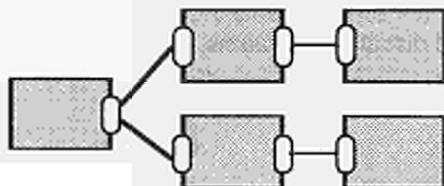


Figure 2 - Systems

In addition to describing overall system structure, Darwin permits systems to be composed hierarchically by allowing the specification of composite components which are configurations of either simple program components or other composite components. The interface to a composite component is specified by the ISL in an identical way to program components (Figure 3). In this way, program components may be replaced by composite components and vice-versa without affecting the rest of the system structure. Finally, Darwin is also used as the language in which to express system extension and modification by structural changes.

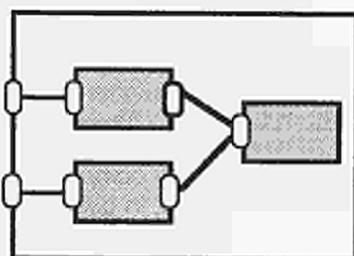


Figure 3 - Composite Component

Section 2 of the paper illustrates the utility and versatility of the configuration oriented approach using a simple telephone exchange as an example. Section 3 discusses the approach, providing a brief comparison with object-oriented programming. Finally, section 4 concludes by summarising the principles of the approach and giving the current status of the work.

2. CONFIGURATION PROGRAMMING IN REX

The main concepts of configuration programming, namely those of explicit structure and hierarchic composition, are illustrated by examples from the REX environment for the development of distributed programs. We concentrate on the configuration facilities provided by Darwin, the REX configuration language. Darwin includes facilities for hierarchic definition of composite components, for parameterisation of components, for multiple instantiation of both components and interface interaction points, for dynamic binding and instance creation, for conditional configurations with evaluation of guards at component instantiation, and even for recursive definition of components. This work owes much to earlier experience using the Conic configuration language

Telephone Exchange Example

Figure 4 illustrates a simple telephone system consisting of a number of telephones interconnected by an exchange. To illustrate the configuration programming approach to developing parallel and distributed systems, we will develop a functional simulation of the telephone exchange using the REX programming system.

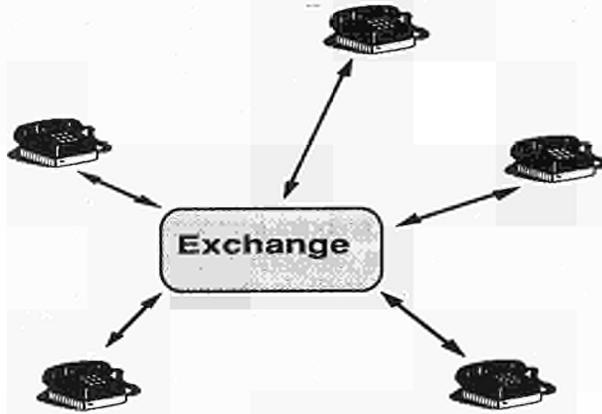
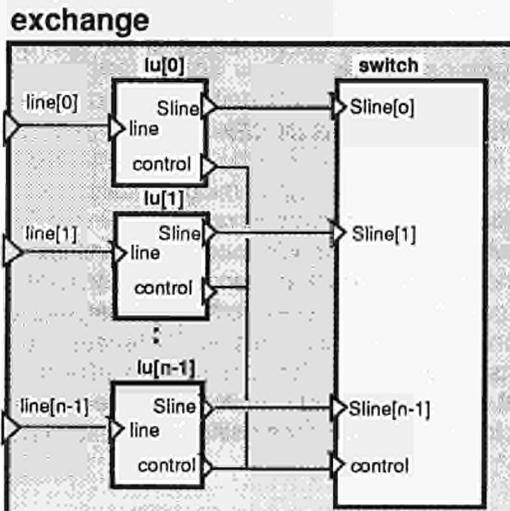


Figure 4 - Simple Telephone system

2.1 Hierarchic component composition

In REX component classes are defined by *component types* which can be instantiated to form components. Figure 5 depicts, in both graphical and textual forms, an initial decomposition of the *exchange* into instances of two component types - lineunits *lu* and a *switch*. Each lineunit handles a subscriber and incorporates the call processing functions to decode control information from incoming signals, return dialling tone, collect digits, and send ring signals. In response to control commands, the switch sets up the simulated speech and signalling paths between the lineunits representing the incoming and outgoing sides of a telephone call.



```

component exchange(n:int) = {
entry
    line:[n] lineT;           // array of ports
use
    exchdefs:lineT;         // port types
    lineunit;               // component types
    switch;
inst
    switch(n);              // switch names both the instance and the type
forall i:0..n-1 {
    inst
        lu[i]:lineunit(i);
    bind
        line[i]           -- lu[i].line;           // bind to interface
        lu[i].Sline      -- switch.Sline[i];      // internal bindings
        lu[i].control    -- switch.control;
    }
}

```

Figure 5 - Exchange Configuration

The program textual description is expressed in the REX configuration language Darwin. The description specifies which component types are used to construct a composite component (**use**), which component instances of these types are created by the composite component (**inst**) and how the ports of these instances are interconnected (**bind**). Component types can have formal parameters which are resolved when the component is instantiated. The exchange component is parameterised by *n* the number of lines it supports. This parameter is used to dimension the array of interaction points **line** and also the array of *lineunit* instances *lu*.

2.2 Interface Specification

In general, the interface to a component is specified in ISL by typed ports. In this case it is the array of **entry** ports *line* of type *lineT*. This type is imported from a file of port type definitions used by the telephone system (listed below):

```

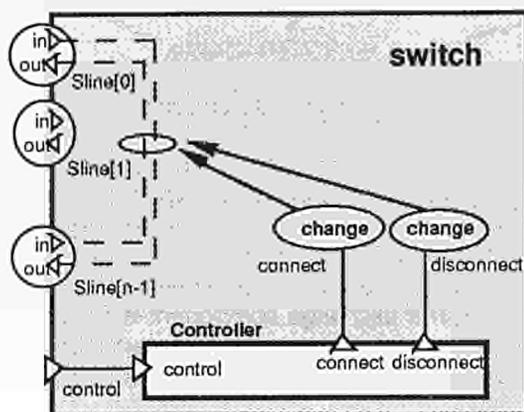
types exchdefs = {
    lineT = {in: port char           // asynchronous
            invert
            out: port char         // asynchronous
    };
    controlT = port int, int,int - int; // synchronous
    Nmax = 32;
    BUSY = 1; OK = 2;
}

```

LineT is defined as a **port set** which consists of two ports of type *char*. These ports have opposite invocation directions indicated by the keyword **invert** which denotes that if *in* is an entry port *out* will be an exit port and vice-versa. Consequently the interface declaration of Figure 5 (**entry** line:[n]lineT) declares an array of port sets in which each set consists of an entry port *in* and an exit port *out*. A component accepts invocations from entry ports and invokes the services of other connected components via exit ports. In the above, *in* and *out* support asynchronous unidirectional invocations ie. the information flow is from the exit port to the entry port and the caller does not wait for a response; whereas *controlT* supports conventional synchronous operation invocation ie. information flow is in both directions and the caller waits for a response.

2.3 Dynamic Binding

The component which implements the exchange switching function is depicted in Figure 6. The switch component includes an example of dynamic binding (and unbinding). In response to control commands, the controller component invokes configuration changes to *connect* and *disconnect* paths between the lineunits indicated by the change parameters. The interface to the lineunits is represented by the port array *Sline*. Changes can be considered to be reconfiguration "methods". They are invoked (and bound to) components in an identical way to programmed methods. In addition to bindings, changes may specify the creation of new component instances and the removal of old instances.



```

component switch(n:int) = {
entry
  Sline:[n] lineT;
  control:controlT;
use
  exchdefs:lineT,controlT;
  controller;
inst
  controller; // controller names both the instance and the type

change connect (incoming: int; outgoing : int) = {
  bind Sline [incoming].out -- Sline [outgoing].in;
  bind Sline [incoming].in -- Sline [outgoing].out
  }
change disconnect (incoming, outgoing : int) = {
  unbind Sline [incoming].out -- Sline [outgoing].in;
  unbind Sline [incoming].in -- Sline [outgoing].out
  }
bind
  controller.control    -- control;
  controller.connect   -- connect;
  controller.disconnect -- disconnect;
}

```

Figure 6 - Switch Component

2.4 Programmed components

Figure 7 shows the program for the programmed component *controller*. Each program component instance executes with its own thread of control. It is an active component or process. The programming language is C to which has been added a component header part, remote invocation (**call**) and invocation handler (**accept**) statements. For example, in figure 7, control commands are handled by **accepting** invocations from the entry port *control*. This is a synchronous invocation. The calling component waits for the invocation to complete. Completion occurs when the called component executes a **reply**. A component may selectively wait on a set of entry ports using a **select** construct similar to that provided in Ada. More details on the invocation primitives can be found in the paper by Magee et al¹⁹. The augmented C is compiled by the REX toolkit into standard C. In addition the toolkit supports heterogeneity and allows for components to be programmed in other languages such as Modula 2.

```

component controller = {
use
    exchdefs; // import all definitions from exchdefs
entry
    control:controlT;
exit
    connect: port int,int;
    disconnect: port int,int;
}

%%c // indicates implementation follows in C

int C[Nmax]; // records connections

main(){
    int incoming,outgoing,operation;
    for ( ; ; ) {
        accept control (operation,incoming,outgoing);
        switch (operation) {
            case CONNECT:
                if ( ! C[incoming] && ! C[outgoing]) {
                    call connect(incoming,outgoing);
                    C[incoming]=outgoing; C[outgoing]=incoming;
                    reply control(OK);
                } else
                    reply control (BUSY);
                break;
            case DISCONNECT:
                call disconnect(incoming,outgoing);
                C[incoming]=C[outgoing]=0;
                reply control(OK);
                break;
        }
    }
}

```

Figure 7 - Controller Component

2.5 Distribution

So far we have not considered distribution of the example. Distribution is achieved by annotating a configuration description with processor allocations using the **@** construct. For example, figure 8 shows the exchange component modified to allocate each lineunit to a different processor. The *processor(i)* part of the **@** clause is a function

which maps the virtual processor numbers to physical processor names or addresses. By default, component instances are created on the same processor as their parent group. Hence, the switch component instance will be created at the same processor where *exchange* is instantiated.

```

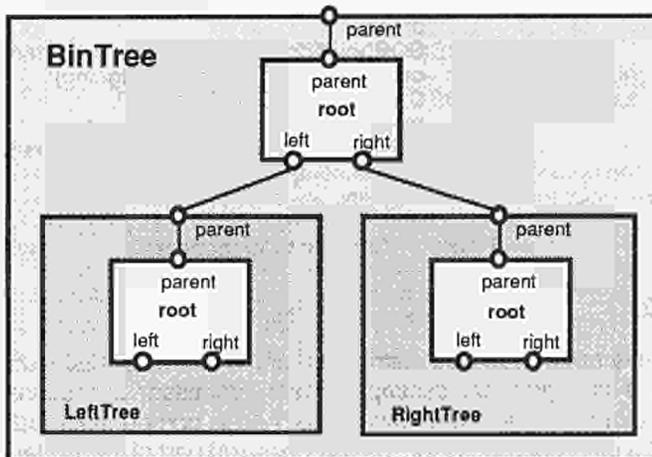
component exchange(n:int) = {
entry
  line:[n] lineT;
use
  exchdefs:lineT;
  lineunit;
  switch;
inst
  switch (n);
forall i:0..n-1 {
  inst
    lu[i]:lineunit(i) @ processor(i);
  bind
    line[i]      -- lu[i].line;
    lu[i].Sline -- switch.Sline[i];
    lu[i].control -- switch.control;
  }
}

```

Figure 8 - Exchange Configuration

2.6 Conditional & Recursive Configurations

The exchange example includes examples of component instance arrays, port arrays and dynamic binding. The following example of a binary tree of component instances demonstrates the features in Darwin to support conditional and recursive configurations (Figure 9). The configuration declarations within the curly brackets following a **when** clause are only evaluated if the **when** condition (in this case *depth1*) is true. *RightTree* and *LeftTree* are both instances of the enclosing type *BinTree* giving the recursive structure definition. Note that in *BinTree* the invocation direction and datatype of the **port** *parent* has not been specified. These can be inferred by the configuration compiler from the component type *node*. This ability to delay the direction and type specification of ports supports top down design in that detail can be added as the design is decomposed into sub-components.



```

component BinTree(depth:int) = {
port
    parent;
use
    node;
inst
    root : node;
when depth1 {
    inst
        LeftTree :BinTree(depth-1);
        RightTree:BinTree(depth-1);
    bind
        root.left -- LeftTree.parent;
        root.right -- RightTree.parent;
    }
bind
    root.parent -- parent;
}

```

Figure 9 - Recursive Binary Tree Component

3. DISCUSSION AND JUSTIFICATION

In this section we discuss and justify the concepts in the REX approach, particularly in relation to Object Oriented Programming (OOP). OOP combines a number of sound and useful concepts in a form which has captured the popular imagination²⁸. *Encapsulation* permits the association of state and behaviour to form an *object*. *Classes* serve as templates from which object instances are created and so permit the reuse of a single behaviour specification for the set of all object instances created from that class. *Inheritance* provides a mechanism for the reuse of the behaviour of one class in the definition of a new class. It establishes a class hierarchy which represents a generalisation / specialisation relationship between classes. All these are very appealing notions. However, there are some weaknesses in OOP and its use for distributed programming.

3.1 Explicit Configuration Structure

We argue that the weaknesses of OOP emanate from the lack of an *explicit description of program structure*. We have shown that a separate and explicit description of program structure facilitates program description, construction and modification, and believe that it complements the object oriented concepts yet is missing from most other OO approaches.

For instance, Object Oriented Design (OOD) methods concentrate on the definition of classes in an inheritance hierarchy. This hierarchy defines the relation between *classes*, but ignores the actual structure of the required program. Actual program structure is poorly expressed, being embedded in the object code. Structure needs to be defined in terms of *object instances*. Explicit program structure, in terms of object instances and the interconnections or bindings between them, is a natural outcome of traditional design techniques such as the system structure diagram of JSD⁶, Data Flow Diagrams and Structure Charts of SASD²². This resulting design structure of object instances can be used to determine the classes required to perform the application processing¹², and can *then* be organised into an inheritance hierarchy to try to maximise software reuse.

To some extent, this lack of an explicit structural description in OOD is excused by the use of dynamic object creation and dynamic binding between objects. However, it is very difficult to determine the current overall structure of a program as it is embedded in the object code as parameters, instructions for instances to be created and references to objects to which bindings are required. We have shown that even such dynamic programs can benefit by making the possible object instance structures more explicit. In addition, structural descriptions provide a clear and useful reference abstraction for the personnel involved in the distributed development process, including program construction, evolution (modification) and software management.

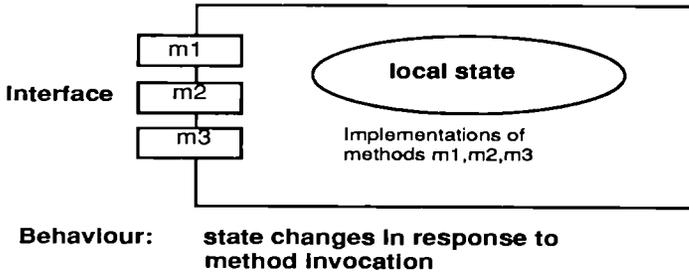


Figure 10. - An Object in OOP

3.2 Interfaces for Context Independence

Modularisation is a necessary facility not only for distribution, but also for sound software engineering. One of its recommended principles is that the interface to a module should explicitly define all the means of affecting its behaviour and state. Let us examine and assess object interfaces in this light.

Objects interact by invoking methods. An object interface is usually described by those methods which it offers (figure 10). However, its use of the methods of other objects is not described explicitly at the object interface but embedded internally in the object code. This is analogous to the publication of offered service interfaces in distributed systems but the hiding of service requirements. We believe that both the "services" *provided* and *required* are necessary for the description of object behaviour and ought to be explicitly defined at the object interface. The interface can thus clearly and explicitly identify the methods offered and used, the types of data associated with each, and even the form of interaction (synchronous or asynchronous invocation).

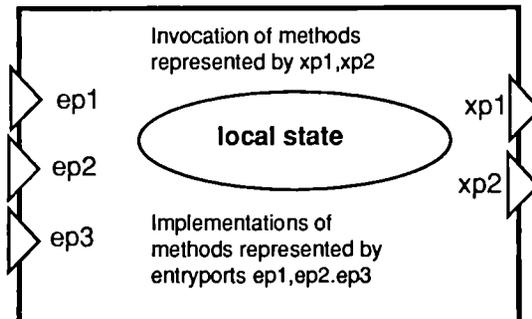


Figure 11 - An Object with explicit interfaces

This symmetry in interface definition can be extended to object naming. In the same way that, from the point of view of the invoked object, an invocation is from an anonymous (unnamed) object, so calls on other objects can be anonymous by making them indirectly via a port. Object code need never refer directly to other objects by reference, but rather indirectly via its port interface. In this way, objects achieve context independence. This facilitates the use and reuse of an object in different environments. Object port binding can then be performed separately. In practice context independence can be achieved by defining an object interface in terms of:

- i) the set of typed *entry ports* representing the methods the object provides,
- ii) the set of typed *exit ports* representing the methods required from other objects (Figure 11).

3.3 Concurrency and Distribution

What of concurrency and distribution? For instance, at what granularity level should concurrency be provided? Should objects become active entities like processes, or should there rather be the ability to create threads of execution within objects?

The latter thread model provides a finer grain of concurrency, but requires the use of additional internal concurrency control primitives (such as semaphores) for synchronisation and to control access to shared data^{21,26}. Use of such synchronisation primitives is generally difficult and error-prone. Although some improvement has been made by permitting separate expression of this synchronisation in systems such as Guide³, they can still require highly complex expressions.

In order to make the concurrency control clearer, the grain of active concurrency can be made coarser and restricted to coincide with an object. This leads us to a more uniform form of concurrency which is clear and well suited to distribution. As illustrated, our approach is the use of sequential active objects (cf. process components) which communicate directly with one another by remote method invocation. A distributed program then consists of a number of distributed communicating objects, which can handle one invocation at a time. This has sacrificed the fine-grain concurrency of the concurrent threads approach, and adopted the object as the grain of concurrency and distribution. In so doing, the need for additional synchronisation primitives within an object are obviated and instead all that is necessary is the ability for an object to wait on a selection of possible invocations/messages. Such active objects are particularly attractive if the conventional synchronous method invocation is extended to include asynchronous invocation.

3.4 Inheritance and Composition

Inheritance in Object Oriented Programming languages permits objects to share both behaviour and data with their parent superclasses. However, while it is easy to see the utility of this mechanism in constructing complex sequential object behaviour, it is less simple in relation to parallel and distributed systems as object encapsulation can be violated. For instance, a subclass may refer to variables or local private operations in its superclass, or even to superclasses further up the inheritance hierarchy. Such use of shared class variables and procedures cannot be efficiently supported in distributed systems. Wegner²⁷ has gone so far as to state that "...*distribution is inconsistent with inheritance*".

On the other hand, composition seems to offer a viable, sensible and efficient alternative to inheritance. Composition is a technique for *constructing* systems, where-

as its inverse, decomposition is a means of *designing* systems. Composition provides a powerful means of abstraction in that it permits a collection of components to be treated as a single component (figure 5). Composition supports reuse in that it permits a new class to be defined in terms of instances of existing classes. Raj and Levy²⁴ also prefer the use of composition; however, in our case, the definition of a composite class uses a separate structural configuration language.

3.4 Summary

The REX approach illustrated in the previous sections is based on the use of active objects (termed *components*), both synchronous and asynchronous method invocation and explicit interfaces and bindings. An explicit configuration language is used to define program structure as a set of components and their bindings. This configuration language, Darwin, is separate from those used for programming components as context-independent types (like classes) with well-defined interfaces. The configuration language includes facilities for hierarchic definition of composite components, for parameterisation of components, for replication of both component instances and interface interaction points, for conditional configurations with evaluation of guards at component instantiation, and even for recursive definition of components. In addition, the approach supports dynamic configuration, expressible at the configuration level as programmed changes to the configuration of component instances and/or their bindings. In addition, evolutionary changes can be similarly expressed but applied interactively.

4. Conclusions

We believe that the combination of OOP concepts with explicit structural description has much to offer in programming distributed systems. Others have also recognised the importance of the structural configuration view in a distributed environment^{1,5,8,10,15,16,17,20,23}.

The REX software architecture has been primarily influenced by the basic principles of configuration programming outlined in^{10,13}. In the following we briefly relate the approach to these principles.

1. *The configuration language used for structural description should be separate from the programming language used for basic component programming.*

REX goes further than this by the use of an Interface Specification language which makes the configuration language Darwin completely independent of the language used to program component behaviour. Darwin may thus be used to construct systems which consist of heterogeneously programmed components.

2. *Components should be defined as context independent types with well defined interfaces.*

The REX ISL specifies both the ports through which remote services are accessed as well as the ports which represent services that a component provides. A component thus makes no direct references to non-local entities. It can thus be used in many different contexts by binding its ports.

3. *Using the configuration language, complex components should be definable as a composition of instances of component types.*

As illustrated in section 2. Darwin provides exactly this. Further, no distinction is made between complex components and basic program components in terms of their use and instantiation.

4. Change should be expressed at the configuration level, as changes of the component instances and/or their connection.

Although REX allows program components to instigate changes to the structure of a system, the changes themselves can only be described in Darwin in terms of instances and bindings.

The REX work is heavily based on Conic, with which we have had extensive experience for a number of years. This has provided us with convincing evidence of the utility of the configuration approach for distributed program design¹², construction¹⁸, evolution¹⁴ and management using graphic tools such as ConicDraw¹¹. However, a number of limitations to Conic and its implementation have also been recognised. It provided support for *post hoc* evolutionary change but no linguistic support for dynamic programmed change, where the configuration changes are embedded in the configuration description (as illustrated in the exchange switch). Component interfaces in Conic are simpler and do not support port sets and the associated binding rules. Finally, there is no integrated support for components written in different programming languages. These limitations are being addressed in REX and its configuration language, Darwin.

The current status of the work is that Darwin has been implemented and the communication primitives have been embedded in components written in C or C++. Colleagues in the REX project are embedding the primitives in other languages such as Modula 2. Management tools are also being developed to monitor and control system configuration. RexDraw (cf. ConicDraw) will provide a graphic facility while an OSL (Object Selection Language)⁷ will provide for the description of generic configuration management. In addition, open systems connection is to be provided by the ability to publish specified interfaces either as services offered or required (figure 12).

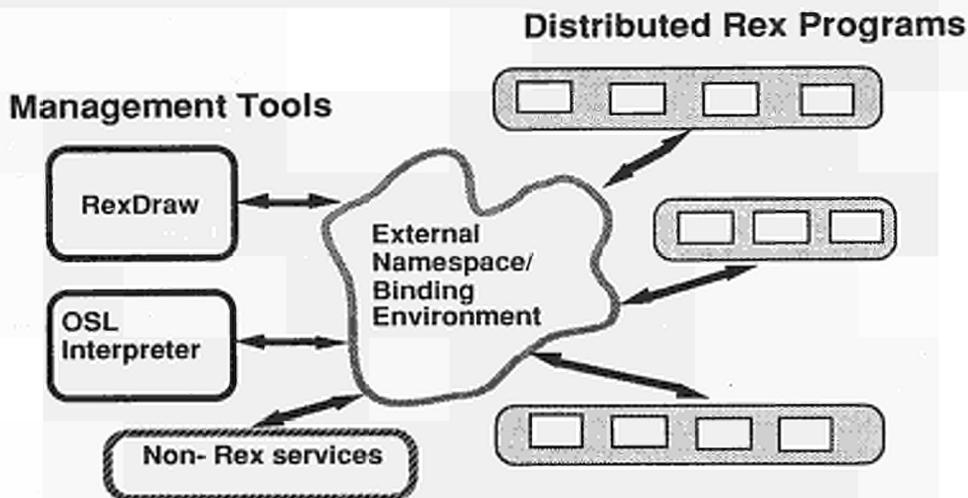


Figure 12 - Distributed Systems Management and Open Systems

As mentioned, the REX project is providing tools to support all phases in the development of reconfigurable and extensible parallel and distributed systems: specification, design, construction, analysis and management. Such tools are being built to exploit the use of a separate structural description of a system. For instance, a prototype design tool, RexDesigner, has been constructed to support the Constructive Design Approach¹², a method which utilises the structural view during the design process.

The REX project offers multiple component programming languages to suit the programming task at hand. Similarly, to provide support for diverse application domains, REX supports multiple specification and analysis techniques and the integration of multiple methods. Specification and method integration is provided through the use of a novel approach, called ViewPoints⁴, which supports specification and method partitioning and interaction through explicit transformations analogous to those used in software configurations. Furthermore, a REX Engineering Environment⁹ is intended to provide a general facility for integrating information produced during any of the phases of software development and maintenance. It provides an attributed file system combined with language facilities for enveloping and controlling the invocation of slave CASE tools and compilers. A generalised attribute mechanism is provided to allow specification and design tools to attach information to configuration descriptions. These attributes, which can include timing and behaviour constraints, can be checked during system construction with the aim of ensuring compatibility between components when their interfaces are bound.

Acknowledgements:

Acknowledgement is made to our colleagues at Imperial College (Anthony Finkelstein, Keng Ng) and also to our partners in the REX project for their contribution to the work described in this paper. We gratefully acknowledge the SERC under grant GE/F/04605 and the CEC in the REX Project (2080) for their financial support.

REFERENCES

- (1) BARBACCI, M.R., WEINSTOCK, C.B. and WING, J.M. (1988). "Programming at the Processor - Memory - Switch Level", Proc. of 10th IEEE Int. Conf. on Software Engineering, Singapore.
- (2) BLACK, A., HUTCHISON, N., JUL, E., LEVY, H., and CARTER, L. (1987). "Distribution and abstract types in Emerald", IEEE Trans. on Software Eng. SE-13(1), 65-76.
- (3) DECOUCHANT, D., LE DOT, P., REVEILL, M., ROISIN, C., and ROUSSET DE PINA, X. (1981). "A Synchronisation Mechanism for an Object Oriented Distributed System", Bull-IMAG, Rapport Technique 9-9.
- (4) FINKELSTEIN, A., KRAMER, J. and GOEDICKE, M. (1990). "ViewPoint Oriented Software Development", Proc. of Third Int. Workshop on Software Engineering and its Applications, Toulouse.
- (5) GOGUEN, J.A. (1986). "Reusing and Interconnecting Software Components", IEEE Computer, (Designing for Adaptability), Vol. 19, 2.
- (6) JACKSON, M.A. (1983). "System Development", Prentice Hall.
- (7) JIAWANG, W. and ENDLER, M. (1991) "A Configuration Model for Dynamically Reconfigurable Distributed Systems". Proc. of 24th HICSS Conference, Hawaii, 265-279.
- (8) KAPLAN, S. and KAISER, G. (1988). "Garp: Graph Abstractions for Concurrent Programming", ESOP '88, Nancy, France, Springer-Verlag, 191-205.
- (9) KOCH, W., NAGEL, K. and OBST, W. (1991). "Guiding System Evolution by Task Engineering Tools", Proceedings of the 5th Conference on Software Engineering Environments (SEE'91), Aberystwyth, Wales.
- (10) KRAMER, J., and MAGEE, J. (1985). "Dynamic Configuration for Distributed Systems", IEEE Transactions on Software Engineering, SE-11 (4), 424-436.
- (11) KRAMER, J., MAGEE, J., and NG, K. (1989). "Graphical Configuration Programming", IEEE Computer, 22(10), 53-65.

- (12) KRAMER,J., MAGEE,J., and FINKELSTEIN,A. (1990). "A Constructive Approach to the Design of Distributed Systems", Proc. of 10th Int. Conf. on Distributed Computing Systems, Paris, 580-587.
- (13) KRAMER,J. (1990) "Configuration Programming - A Framework for the Development of Distributable Systems", Proc. of IEEE COMPEURO'90, Tel-Aviv, Israel, May 90, pp 374-384.
- (14) KRAMER,J., and MAGEE,J., (1990). "The Evolving Philosophers Problem: Dynamic Change Management", IEEE Transactions on Software Engineering, SE-16 (11), 1293-1306.
- (15) LEBLANC,R.J. and MACCABE,A.B. (1982). "The Design of a Programming Language based on a Connectivity Network", Proc. 3rd Int. Conf. On Distributed Computing Systems.
- (16) LEBLANC,T. and FRIEDBERG,S.. (1985). "HPC: A model of structure and change in distributed systems". IEEE Trans. on Computers, Vol. C-34, 12.
- (17) LEE,I., PRYWES,N. and SZYMANSKI,B. (1986). "Partitioning of Massive/Real-Time Programs for Parallel Processing", in Advances in Computers, ed. M.C. Yovits, Vol.25, Academic Press.
- (18) MAGEE,J., KRAMER,J., and SLOMAN,M. (1989). "Constructing Distributed Systems in Conic" IEEE Transactions on Software Engineering, SE-15 (6).
- (19) MAGEE,J., KRAMER,J., SLOMAN,M. and DULAY,N. (1990). "An Overview of the REX Software Architecture", Proc. of 2nd IEEE Computer Society Workshop on Future Trends of Distributed Computer Systems, Cairo.
- (20) NEHMER,J., HABAN,D., MATTERN,F., WYBRANIETZ,D. and ROMBACH,D. (1987). "Key Concepts of the INCAS Multicomputer Project", IEEE Transactions on Software Engineering, SE-13 (8).
- (21) NIERSTRASZ,O.M. (1987). "Active Objects in Hybrid", Proc. OOPSLA '87, Sigplan Notices, Vol 22, No 12, 243-253.
- (22) PAGE-JONES,M. (1988). "Practical Guide to Structured Systems Design", Prentice Hall International Editions.
- (23) PURTILO,J. (1988). "A Software Interconnection Technology", Computer Science Dept., University of Maryland, TR-2139,.
- (24) RAJ,R.K. and LEVY,H.M. (1989). "A Compositional Model for Software Reuse", The Computer Journal, 32 (4), 312-322.
- (25) REX Technical Annexe (1989), ESPRIT Project 2080, European Economic Commission.
- (26) STEIGERWALD,R. (1990). "Concurrent Programming in Smalltalk-80", ACM Sigplan Notices, Vol 25, No 8, 27-36.
- (27) WEGNER,P. (1987). "Dimensions of Object-Based Language Design", Proc. of OOPSLA '87, 168-182.
- (28) WEGNER,P. (1990). "Concepts and Paradigms of Object-Oriented Programming", OOPS Messenger (ACM SIGPLAN), Vol. 1, NO. 1, 7-87.

ADVERSE - ENVIRONMENT RECOGNITION OF SPEECH (A.R.S.)

Giancarlo Babini (CSELT), Luciano Fissore (CSELT)
William Ainsworth (Keele University), Eleftherios Frangoulis (Logica Cambridge)
Ramon Garcia Gomez (Etsit-Upmad), Yves Grenier (Enst-Arecom)
Philippe Lockwood (Matra Comm.)

CSELT - Centro Studi e Laboratori Telecomunicazioni
Via G. REiss Romoli 274 - 10148 Torino, Italy
Telephone: +39 11 21691
Telex: 220539
Telefax: +39 11 2169520

Abstract

ARS is a R&D Project aiming at extending the state of the art in speech recognition and placing this innovative technology in adverse environments such as car and factory floor, according to the interests of the Partners.

Starting from an established base of expertise of each Partner, this Project aims at the theoretical work on algorithms and at the development of hardware prototypes. To get the best recognition performance, algorithms covering the different aspects of signal processing were considered. The activities were subdivided into 6 work packages concerning respectively system definition and standards, transducers and noise reduction, feature extraction, pattern processing, human factors and user interface, system prototyping and evaluation.

After a brief presentation of the general structure of the Project (objectives, organisation, participation, resources), this paper will present the state of the work after two years [1]. For each Work Package the results obtained, the work in progress and the activities planned for the next months will be briefly presented.

1. Introduction

Automatic speech recognition may be considered an innovative technology on the border between the research field and the application field: high performance speech recognition systems are available in many research laboratories, but they need to be adapted to the particular constraints imposed by the applications.

Factory and car housing were the environments considered in the Project; in both of them, and mainly in the latter, the background noise conditions may be very high and widely varying according to the surroundings; moreover the stressing circumstances may alter the user's behaviour and his/her voice characteristics.

Therefore, the first aim of the Project was to examine thoroughly the drawbacks encountered in the practical applications and, on the established base of expertise of Partners, to study and evaluate several algorithms to select the more suitable ones.

A second aim was the development of a hardware platform common to all the participant partners, giving the facilities of exchanging with little problem the software modules produced during the Project development and to realize prototypes integrating the selected algorithms.

In the paper, after a short description of the Project, the state of the activities at the end of the second year will be presented, together with the most significant results obtained until now. Then an outline of the further developments will be given.

2. Project general outline

ARS Project has a three years duration, starting on March 16th 1989 and ending on March 1992, with a man power effort of 47 man years. There are nine Contractors and an Associated Contractor participating to the Project, from four countries: England, France, Italy and Spain. The activities were subdivided into the following six workpackages:

- WP1: System definition and infrastructure;
- WP2: Transducers and signal preprocessing;
- WP3: Feature extraction and distance measures;
- WP4: Pattern processing;
- WP5: Human factors and user interface;
- WP6: System prototyping and evaluation;

In Fig. 1 the development outline of the Project is shown.

Work Package 1 was started at the beginning of the activities and was concluded after three months as an essential item for the further collaboration among partners and as a base to achieve a final single common recognition system. Indeed, starting from different speech recognition systems already existing in the partners' laboratories, a definition of a common system and a standardisation of tools were mandatory to get a final modular structure, so that the individual works can be easily compared and integrated.

Soon after that, the studies of algorithms in workpackages WP2, WP3, WP4, WP5 started; the end of these studies was set at the end of the second year when the evaluation and the selection were carried out.

In WP6, at the beginning of the Project, a first data base was collected in the field environment (factory and car) for four languages (English, French, Italian and Spanish) to be used for training and testing. The data base was used for testing the recognition systems owned by the partners with the aim of getting a reference in terms of performance from which to compare the results obtained through the refinements of the techniques.

Moreover, in WP6, the study of the common hardware structure was started and, later in the second year, the prototype building was started too, and it will be completed in the third year.

The third year of the Project will be mainly devoted to the integration of the algorithm software modules into the real time demonstrator and for the final system evaluation in laboratory and in the field, by exploiting the data base obtained by merging the first one (collected at the beginning of the project) and the complementary second one which was gathered according to the specifications provided by WP2.

Moreover, the study and the integration of the user interface will be completed before the middle of the third year.

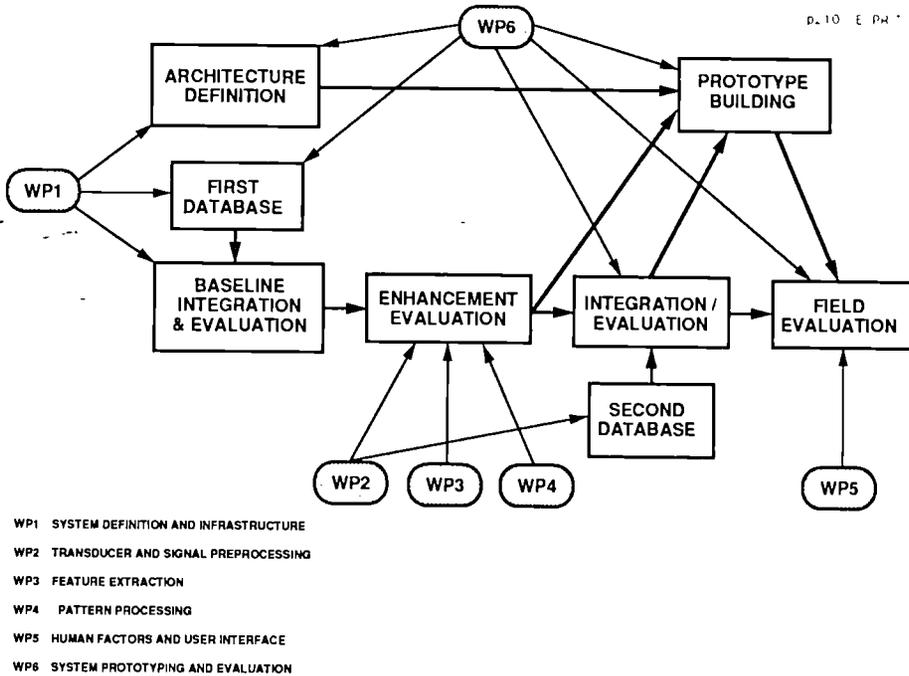


Fig. 1 ARS Project Outline

3. The recognition system

The functional block scheme of the recognizing system is reported in Fig. 2. It is a speaker dependent recogniser for a vocabulary ranging from 50 to 500 words; a user interface with voice response is provided to address the problem of man - machine dialogue in a practical application.

3.1 Functional structure description

The function of speech preprocessing is to pick up the speech signal from the environment reducing the effects of noise: this goal may be achieved with different techniques used singularly or in a parallel way.

The approaches investigated were:

- an array of microphones, by which noise reduction can be achieved with the realisation of an adaptative beamformer that focuses the speech source;
- use of a two-microphones approach to exploit the spatial incoherence of noise with respect to the coherence of speech;
- single channel noise cancellation by means of linear and non linear filtering;
- noise subtraction in the frequency domain based on spectral noise model or short time modified coherence representation (SMC).

Other important points to be considered are the kind of microphone which should be used and be evaluation of the best positions of microphone installation as regards the user position and the noise sources.

All these techniques were investigated in Work Package 2.

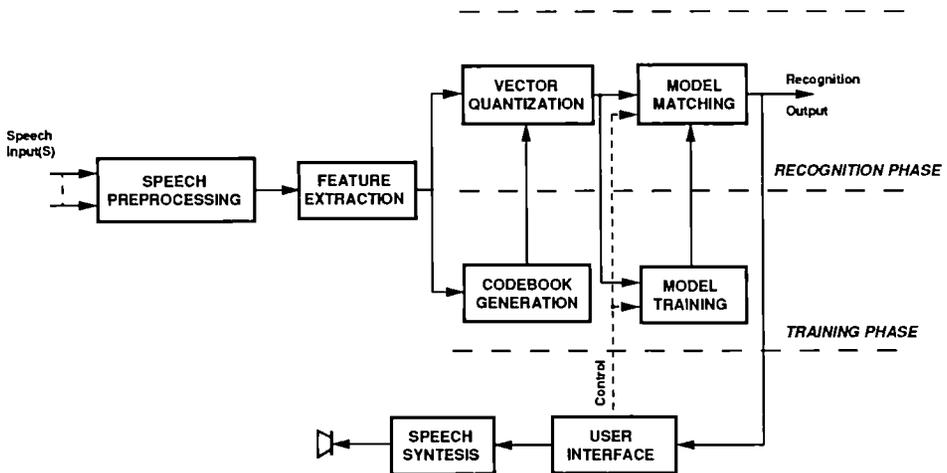


Fig. 2 Speech Recognising system functional blocks

A second functional block, very important for the recognition performance is represented by the feature extraction which generates the reference knowledge about the acoustic units to be recognised. Several techniques have been examined and evaluated; among them the following can be highlighted:

- Parametric representation:
 - Linear Predictive Coding (LPC) and its derivatives as LAR (log Area Ratio), Kr coefficients and LSP (Line Spectrum Pair);
 - weighted Cepstral coefficients computed from LPS (LPCC);
 - Short-time Modified Coherence.
- Non parametric representation:
 - Mel Frequency Spectrum (MFCC) [2] computed by a DCT from the outputs of a filter bank;
 - weighted MFCC with different types of filters (inverse global variance, spectral peak adjustment, bandpass filter);
 - Delta MFCC, so as to include temporal information
- Perception-based representation:
 - physiological model, simulating the impulse response of nerve fibers;
 - design of blocks to approximate the function of the auditory system without observing the nerve response.

Strictly correlated to the feature extraction methods are the distance measures techniques (Euclidean in frequency space, Mahalanobis, Root Power Square, sine lifter, etc.), that were also investigated aiming at the achievement of the best performance of the pattern matching module.

The activities on feature extraction and distance measure were carried out in Work Package 3.

The pattern matching function is the heart of the recogniser system and a major effort was devoted to this module, to achieve robustness to noise and high performance.

From the beginning of the Project the technique accepted for the pattern matching was that based on Hidden Markov Model (HMM) [5], as it reaches remarkable performance and allows further developments for speaker adaptation, evolution towards speaker independence and continuous speech recognition.

In the HMM framework, the aim of the Project was to build statistical models of the words, as much as possible insensitive to the noise and its variations. In the following several of the most significant items are reported:

- explicit noise modelling and definition of a method for explicit and dynamic noise compensation (noise adaptation);
- analysis of filtering strategies for noise cancellation within HMM;
- study of training algorithms accounting for the application constraints [3, 4];
- choice of a smoothing technique to overcome the statistical insufficiency of data;
- study of duration modelling to capture informations about the duration of sounds;
- use of multi-speaker models to achieve a relative speaker independence;
- adaptation to new speakers and to varying environments [18].

The activities on HMMs were carried out in the Work Package 4.

To complete the recognition system, the user interface is the functional module, which allows the exploitation of the recognition outputs. A careful design of the Man-Machine Interface (MMI) lets the usability of the system to be increased, which must be friendly and easy to use [14, 15]. The training procedure must be easy, and a user guide system is needed.

The study of the user interface was carried out in the Work Package 5 and dialogue and training procedures were carried out in order to be implemented, after the evaluation, on the final demonstrator.

3.2 Common Tools

The assessment of the various algorithms and the exploitation of the synergy among the Project Partner's efforts called for a common definition of the interfaces among the functional modules as shown in Fig. 2; indeed only in that case is an easy interchange of software packages and speech data produced possible.

At the very start of the ARS Project, the final report on the Extension Phase of SAM Project (ESPRIT Project no. 1541) was released: ARS partners judged the SAM assessment methodology to be well founded and its recommendations were followed. The standards defined within the project were set according to SAM philosophy.

In particular, the speech sample files (at the input of the speech preprocessing and of the feature extraction modules) and the results (at the output of the recognition module) were stored according to the file formats; new formats were derived for the specific requirements of ARS in the output of the feature extraction stage and of the codebook generation modules. The output format of the model training module is strongly correlated to the technique adopted; so a standardisation was not possible at the beginning of the project.

The adoption of SAM formats lets the test of ARS recogniser be made by using SAM databases and the test of commercial or research recognisers through ARS databases. This is very valuable for both projects.

A fundamental tool for speech recognition training and testing is a large sized data base. Two data bases were collected (the second complementary to the first one) by the different partners for four languages. Then a multilingual data base (at sample level) was extracted and distributed among the partners, to be used as a common training and testing tool for performance comparison of different algorithms.

The multilingual database was also provided in terms of spectral features as the output of a standard front-end-processor already used in the Esprit Project P26. These data are very useful for performing a cross evaluation of different training strategies.

Finally, a baseline HMM system has been implemented in compliance with the specifications defined in WP4; this system allows a set of experiments to be carried out to compare and assess the performance of the enhanced modules of the recognition system.

3.3 Hardware structure

A common hardware structure was devised by the Project partners for the realisation of the final real time demonstrator and of the prototypes for the field test. Already at the beginning of the project the use of PC or compatible devices with PC speech processing board was stated, although other devices could be used in laboratory. The main programming languages were stated to be C and FORTRAN.

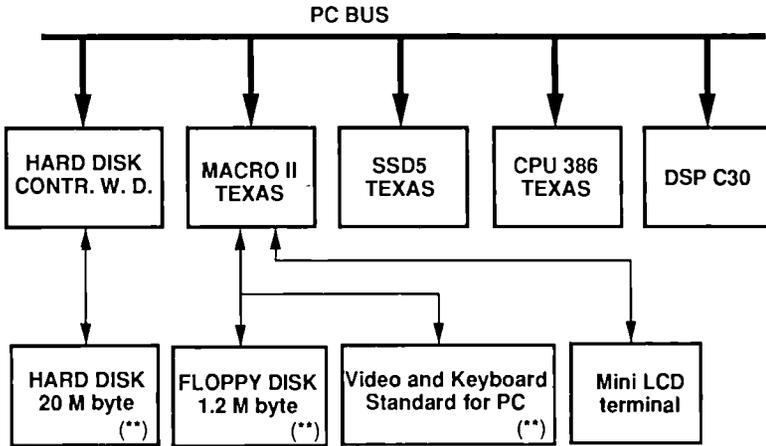
Those statements defined a common base for the realisation of the prototyping standardisation, which were carried out exploiting the latest family of DSP chips [13].

In Fig. 3 the hardware structure configuration of the real time recogniser system is shown. It uses off-the-shelf floating point DSP boards in a PC environment, for which a transportable version may be used for field test; moreover the TMS320C30 Digital Signal Processor was selected.

This configuration can be used in the car and the laboratory; the laboratory configuration is a superset of the car configuration.

5 standards PC modules are connected together with the PC bus. They perform the following functions:

- the first block in the Winchester hard disk controller which is used together with a 20 Mbyte hard disk in laboratory only;
- the Macro II is a multifunction board which controls the floppy disk, the standard PC keyboard, the videodisplay and the supplementary LCD terminal used in the car;
- the SSD5 board is a 2.5 MBytes solid state RAM used as a hard disk in the car environment; the memory is modular with 256 kbytes modules;
- the central processor unit is functionality equivalent to the IBM AT computer; its major features are:
 - 80386 main processor with a 20 MHz clock;
 - 4 Mbytes RAM memory on board;
 - 80387 mathematic co-processor;
 - set up program stored in ROM.
- the last board will contain the digital signal processor and the AD and DA converter; this board can be duplicated if necessary to cover the computing requirements for algorithms;
- the mini LCD terminal has a 16 characters by 4 lines display and a 16 keys keypad; it will be used in the car for training and field testing.



(**) For the laboratory configuration only

Fig. 3 Prototype Hardware Structure

4. Review of the major results obtained in the algorithm studies

There are several items that have to be carefully approached and evaluated during the design of a speech recognition system operating in a noisy environment. Among them, the most significant ones which have been addressed in the workplan of the project with promising results, are summarised in the following.

The first aspect is the proper choice of a noise-cancelling microphone for recording in the car environment [9] and the use of an adaptative microphone array [12]. After a careful comparison performed by using 11 microphones, the uni-directional Audio-Technica 831-H7 back electret microphone, characterized by a 3 db-cutoff at 10 MHz has been selected. Among the possible positions of a hand-free mike in the car, 2 of these have been chosen: left upright and steering wheel.

Another significant item addressed by the Project partners has been the use of speech enhancement algorithms to increase the SNR [6]. The adaptative noise cancellation by non-linear filters and a noise-reduction algorithm relying upon a single microphone, or a couple of microphones, have been chosen for more careful evaluations currently in progress. In particular, a frequential approach based on a non-linear spectral subtraction (NSS) performs very well, as it allows a very significant improvement of the performance of the overall system in terms of correct recognition rate, especially for very noisy data [19,20].

Table 1 and Table 2 show a comparison among the performances obtainable by using Italian and French databases (car moving at 90 Km/h and 130 Km/h) with and without this speech enhancement algorithm. In particular, the columns with the tag MFCC refer to the use of Mel scale Cepstral Coefficients while the ones with NSS are for the speech enhanced case. The vocabulary is made up of 34 words for Italian and of 43 for French. The training session is carried out with the car standing (10 repetitions for Italian, 4 repetitions for French) by using the Baseline HMM training algorithm defined in the project. Five repetitions for Italian and 4 repetitions for French were used to test the recogniser for each of testing conditions. These results highlight that the improve-

ment of the subjective quality of the speech reflects itself in an appreciable improvement of the correct recognition rate.

	MFCC	NSS	MFCC	NSS
speaker	90Km/h	90Km/h	130Km/h	130Km/h
CR	100	100	74.1	100
EB	99.4	100	91.2	99.4
MB	99.4	99.4	91.2	98.8
RC	100	100	98.2	100
average	99.7	99.85	88.67	99.55

Table 1: Correct rate (%) for Italian database with and without speech enhancement

	MFCC	NSS	MFCC	NSS
speaker	90Km/h	90Km/h	130Km/h	130Km/h
GI	98.8	100	88.9	100
LA	96.5	100	96.5	100
CO	79.1	98.8	41.9	96.5
CH	95.9	99.4	60.5	99.4
Average	92.6	99.6	71.9	99.0

Table 2: Correct rate (%) for French database with and without speech enhancement

The design of a feature extractor providing a set of suitable parameters in presence of speech degraded by noise [1] is a significant objective of the project. Several different algorithms have thus been evaluated during the first phase of the project. After some preliminary tests, carried out by using a subset of the multilingual database, MFCC (Mel-Scale Cepstrum Coefficients) and SMCC (Short Time modified Coherence) have been selected as non-weightings and distance measures is in progress by using the baseline HMM system and the whole speech data.

Another objective of the project is the design of speech recognition models and algorithms which are robust enough to achieve high performance even in adverse environments [10]. The selection of a suitable training procedure has been performed by the Project partners accounting for several factors such as algorithm computational complexity and memory requirements, size of the available training database, training conditions, user interface, and accuracy of the obtained models.

To convert laboratory researches into a working prototype, constrained by the application, particular attention has been devoted to the user interface. The first constraint that must be tackled is the amount of training data that must be necessarily small as it would be awkward for the user to utter many replications of the application vocabulary. A second constraint, in a car environment, suggests that the training session be performed with the car standing, for the sake of user friendliness and safety. The speech recogniser will then be used in different conditions (car running at different speeds) and the input will be affected by various and varying sources of noise.

Among the different techniques evaluated by the Project partners, the Fixed Variance (FV) Training algorithm [8,7] has been selected. The performance of this training method, that does not require the reestimation of the variances, has been compared with respect to the classical Forward-Backward (FB) algorithm in a preliminary set of experiments carried out on the Italian database. The results of this comparison given

in Table 3 show that the Fixed Variance method consistently outperforms the FB one, particularly in noisy conditions.

SPEAKER	FV		FB	
	90Km/h	130Km/h	90Km/h	130Km/h
CR	100	88.8	100	83.5
EB	100	88.2	100	83.5
MB	99.4	87.1	98.8	86.5
RC	100	97.7	100	91.2
Average	99.8	90.4	99.7	86.2

Table 3: Correct rate (%) for Italian database - Fixed Variance and Forward Backward algorithms

Another topic that requires careful consideration in a speaker dependent system to be trained in a car environment is the computational complexity of the training algorithm. A Viterbi heuristic model estimation procedure has been devised that presents a remarkable complexity reduction with respect to the FB algorithms. The algorithm is also attractive because it requires a small amount of memory: just the current word model and its most recent application must be stored.

FB algorithm and Incremental Viterbi Training (with Fixed Variance) give comparable results in terms of recognition rates for Italian and English database (see Tables 4 and 5). The English database (collected in the factory floor) is made up of 30 words (digits and command words) by 4 speakers, with 10 clean repetitions and 10 repetitions with additional noise produced by random hammering of metal and operation of an electrical drill close to the speaker.

SPEAKER	FV				Incremental Training			
	90Km/h		130Km/h		90Km/h		130Km/h	
	5 rep	10 rep	5 rep	10 rep	5 rep	10 rep	5 rep	10 rep
CR	98.5	100	90.6	88.8	100	100	88.3	87.7
EB	100	100	91.2	88.2	100	100	89.4	88.3
MB	98.8	99.4	84.7	87.1	98.8	98.8	83.6	85.3
RC	99.4	100	94.7	97.7	99.4	99.4	94.7	97.1
Average	99.2	99.8	93	90.4	99.6	99.6	89	89.6

Table 4: Correct rate (%) for Italian database - Fixed Variance and Viterbi Incremental Training

To increase the noise robustness of the HMM recognition system, several noise compensation algorithms have been considered and [16, 17] evaluated. So far, it has not been possible to select a technique which outperforms the other ones. Further investigations, with a larger database, are needed, as this topic is meaningful even in the case of presence of speech enhancement algorithm.

In the real application, the recogniser must be ready to deal with words that do not belong to the application vocabulary, and its performance may be affected by extraneous sounds such as clicks, external noises, breath noises, hesitations. Several methods for rejecting either mis-pronounced or extraneous words have been investigated. The design of an "average model" of speech, in conjunction with a score-based independent data collected in various noise conditions.

SPEAKER	FV				Incremental Training			
	Hamm		Drill		Hamm		Drill	
	5 rep	10 rep	5 rep	10 rep	5 rep	10 rep	5 rep	10 rep
MP	100	100	100	99.3	100	100	98	99.3
RW	96.7	97.4	96	96	96.7	96.7	94.7	96.7
VB	99.4	99.4	-	-	99.4	99.4	-	-
VS	99.4	100	96	96	98.7	100	96	95.4
Average	98.9	99.2	97.3	97.1	98.7	99	96.2	97.1

Table 5: Corret rate (%) for English database - Fixed Variance and Viterbi Incremental Training

It is worth noting that this review does not cover all the algorithms studied during the first two years of the project.

This section has been devoted to those techniques which, after a careful evaluation, assure an appreciable improvement of the performance, complying with the application constraints. Several of these algorithms have a good chance of being inserted into the final prototype.

However, further evaluations must be performed, by using the extension of speech data base, whose collection is just finished, in order to have a more reliable assessment of the different techniques in terms of statistical consistency.

5. EVOLUTION OF THE ACTIVITIES

During the third and last year of the ARS Project the objective of a common real-time recogniser prototype implementation will be pursued.

During the first quarter the evaluation of the separate algorithm modules, already in progress, will be completed to get a comparative assessment of the different techniques and to consolidate their selection using a larger voice data base for testing.

Soon after that, the integration of the selected algorithms into a general purpose computer will be carried out for a global evaluation of the recognition system, which will be comprehensive of training facilities and of user interface for the car environment.

In the meanwhile, the real-time prototype hardware and firmware will be realised during the third quarter, in such a way to be able to start the field testing which will be completed by the end of the Project. In order to comply with this task, the prototype will be provided of the user interface for car application.

A marginal effort will be spent to introduce further refinements in several algorithms, whose studies are currently in progress, like: speaker adaptation, noise cancellation using the SMCC technique combined with the two channel approach, non-uniform segmentation, custom vocabulary recognition, etc. The results of these activities will be available by the end of the third quarter and will be introduced into the final demonstrator only if there will be man power availability; otherwise they will be included in the following system exploitation.

It is worth noting that, at the end of the Project, a single system will be available, which will include the best investigated solutions, available without incompatibilities by all the Partners of the Project.

6. Conclusions

As it is pointed out in the above presentation of activities, a large effort was devoted, in the two years of the Project, to the study of the algorithms and to their evaluation.

A large number of possible technical choices were considered for the signal preprocessing, the feature extraction and the pattern processing with the aim of selecting the best techniques for the recognition in noisy environments.

The selection of the algorithms to be installed into the final demonstrator of the Project, has required a considerable effort for testing and comparing the different techniques and for analysing the results. Moreover some common tools had to be prepared to get comparable results.

Although we can consider that the studies on algorithms have fulfilled the foreseen objective to get enhanced performance in noisy environments, at the editing time of this paper, the evaluation for selecting the modules more suitable to be integrated into the final demonstrator is still in progress.

It is worth noting that the evaluation procedure has been conditioned by some trade-offs associated with the application constraints; it requires an accurate statistical analysis able to verify the confidence level of the obtained results.

As far as the Project demonstrator is concerned, which will be the most significant output of the Project, the first keypoint was the decision to realize the system on the same hardware using the same DSP, C programming language, and, when possible, also the same boards on the PC system or compatible.

The second one was that the demonstrator will be unique, by means of the integration of the more promising software modules, according to the evaluations in progress; this decision allowed the share of the activities among the Partners.

Other activities very important for the development of the recogniser system are the realisation of the man machine interface and the large sized data base collections.

In conclusion, it may be pointed out that the final objective of getting a noise robust and efficient recognition system is in the consolidation phase and a complete system will be available by the end of the Project time.

Acknowledgments

The authors would like to thank all the staff of the Project Partners for the work performed during these two years.

References

- [1] ESPRIT II Project No. 2101 - ARS **PERIODIC PROGRESS REPORT No.2. Period from the 16th of March 1990 to the 15th of March 1991.**
- [2] K.H. Davis and P. Mermelstein. **Comparison of parametric representation for monosyllabic word recognition in continuously spoken sentences.** *IEEE Trans. Acoust., Speech and Signal Processing*, 28:357-366, 1980.
- [3] L. Fissore, M. Codogno, and G. Pirani. **Isolated word recognition in the mobile-radio system, experiments and results.** In *Proc. of EUSIPCO*, Barcelona, Spain, 1990.
- [4] L. Fissore, P. Laface, M. Codogno, and G. Venuti. **HMM modelling for voice-activated mobile-radio system.** In *Proc. of ICSLP*, Kobe, Japan, 1990.
- [5] B.H. Juang and L.R. Rainer. **Mixture autoregressive Hidden Markov Models for speech signals.** *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-33(6):1404-1413, 1985.
- [6] J.S. Lim. **Speech Enhancement.** Prentice-Hall, 1983.

- [7] R.P. Lippmann, E.A. Martin, and D.B. Paul. **Multistyle training for robust isolated-word speech recognition.** In *Proc. of the ICASSP*, 1987, sect.17.4.
- [8] E.A. Martin, R.P. Lippmann, and D.B. Paul. **Two-stage discriminant analysis for improved isolated word recognition.** In *Proc. of the ICASSP*, 1987, sect. 17.5.
- [9] J.R. Sank. **Microphones.** *J. Audio Eng. Soc.*, 33(7/8):514-547,1985.
- [10] J.G. Wilpon and L.R. Rabiner. **Application of hidden Markov models to automatic speech endpoint detection.** *Computer Speech and Language*, 2(3/4):321-341,1987.
- [11] L. Sanchez-Sandoval, J. Gomez-Mena and R. Garcia Gomez. **A comparative study of feature extraction methods for noisy speech recognition.** In *Proc. of EUSIPCO*, Barcelona, Spain, 1990.
- [12] Y. Grenier, M. Xu, J. Prado and D. Liebenguth. **An adaptative microphone array for speech input in cars.** In *Proc. of ISATA '90*, 1990.
- [13] I.P. Leslie and D.L. Levett **Real-time speech recognition using DSPs.** In *Proc. of ISATA '90*, 1990.
- [14] W.A. Ainsworth and S.R. Pratt **Feedback strategies for error correction in speech recognition system.** In *International Journal of Man-Machine Studies*, 1991.
- [15] W.A. Ainsworth and S.R. Pratt **Voice activated telephone dialogue for digit dialling with error correction.** *SALT Workshop on Speech and Language Processing*, 1990.
- [16] V.L. Beattie and S.J. Young **Noisy speech recognition using hidden markov model state-based filtering.** In *Proc. of the ICASSP*, 1991.
- [17] E. Frangoulis **Isolated word recognition in noisy environment by vector quantization of the HMM and noise distributions.** In *Proc. of the ICASSP*, 1991.
- [18] V. Sgardoni and E. Frangoulis **A novel speaker adaptation approach for continuous densities HMM's.** In *Proc. of the ICASSP*, 1991.
- [19] C. Baillargeat, J. Boudy and P. Lockwood **Noise reduction for speech enhancement in cars by Kalman filtering.** In *Proc. of the EUROSPEECH*, 1991.
- [20] P. Lockwood, J.M. Gillot, J. Boudy and G. Faucon **Experiments with a non linear spectral subtractor (NSS) for speech recognition in the car.** In *Proc. of the EUROSPEECH*, 1991.

DELTA-4 ARCHITECTURE VALIDATION¹

K. Kanoun^{*}, J. Arlat^{*}, L. Burrill^{**}, Y. Crouzet^{*}, S. Graf^{***}
E. Martins^{*}, A. MacInnes^{**}, D. Powell^{*}, J.-L. Richier^{***}, J. Voiron^{***}

^{*} LAAS-CNRS

7, Avenue du Colonel Roche
31077 Toulouse - (F)

^{**} SRD-AEA Technology

Wigshaw Lane, Culceth
Waffington WA3 4NE - (UK)

^{***} LGI-IMAG

B.P. 68
38402 Saint Martin d'Hères - (F)

SUMMARY

The Delta-4 architecture features original fault tolerance concepts for the development and design of dependable open distributed computing systems. To tackle the complex dependability validation task, a comprehensive strategy incorporating both objectives of the validation process - *fault removal and fault forecasting* - has been followed that is based on the use of complementary validation methods. The methods considered include *formal verification* of the design of two specific reliable communication protocols, experimental validation of the implementation of the fault tolerance mechanisms by means of *physical fault injection and analytical modeling and evaluation* of hardware and software. The paper summarizes the main characteristics (objectives, implementation and results) for each of the methods applied in this validation strategy.

1. INTRODUCTION

The validation of a complex fault-tolerant distributed architecture such as Delta-4 is a truly difficult task. Currently, no single validation technique offers a sufficient level of confidence to address alone the dependability validation issue. Accordingly, as was already recognized in previous studies of complex fault-tolerant architectures (e.g. SIFT [SM83], MAFT [Walter90]) the coordination of several complementary validation methods - formal and experimental - is actually needed.

It is worth noting that, in the validation process, we have focused on the most innovative and/or critical parts of the architecture [G90050]. The fault tolerance features of the Delta-4 architecture are based on a multicast communication system (MCS) that provides generalized multicast services. The proper operation of MCS is further based on:

¹ This work was performed within the framework of the ESPRIT II Project P2252 Delta-4, 'Definition and Design of an Open Dependable Distributed System Architecture'.

- well defined properties that characterize the services provided by specific innovative protocols, such as **Atomic Multicast protocols (AMp)** or **Inter Replicate protocols (IRp)**,
- the assumed **fail-silent** property (a kind of halt-on-failure behaviour) of the underlying **network attachment controllers (NACs)** hardware modules that connect the host computers to the Delta-4 network.

The validation of the Delta-4 architecture that is described in this paper is based on a comprehensive strategy incorporating both objectives of the validation process, i.e. *fault removal* and *faultforecasting* [Laprie89].

Removal of design and implementation faults is obtained out by means of verification at two levels:

- **formal verification** of the design of the communication protocols in order to detect and correct software design errors,
- **experimental verification** of the implementation and of the effectiveness of the selfchecking and fault-tolerance mechanisms by means of injection of hardware faults.

Fault forecasting is carried out by means of dependability evaluation, i.e. the estimation of the presence, the creation and the consequence of faults. Dependability evaluation addresses the hardware as well as the software levels. Towards this end, Markov-based modeling and evaluation activities are carried out on the basis of a hierarchical system decomposition that is used for incremental construction of a global dependability model.

For the verification of both protocols (AMp and IRp), we apply **formal design verification** methods based on model checking techniques [CES83, FRV86].

AMp supports reliable distributed communication between groups of entities, be they logically distinct application software components or replicates of software components that have been installed on distinct hosts for the purpose of fault tolerance. IRp is more specifically concerned with the error processing strategy that determines the coordination of replicate communication activity in order to mask errors due to faulty replicates.

The usual informal specifications given in an implementation guide are not sufficient to achieve this aim. The first task is thus to provide a structured formal description of the protocols and of the delivered services, the second one to formally verify the consistency of the service definition with this protocol description. These formalizations and verifications are useful in their own right, since they help to detect inconsistencies of the formal specifications early in the development activity. The software is derived from this specification.

Due to the inherent complex behaviour of a distributed system in the presence of faults **fault injection** is particularly attractive in order to investigate a large number of faulty behaviours [CB89, GKT89, CS90, Walter90]. By speeding up the occurrence of errors and failures, fault injection is a method for testing the fault-tolerance mechanisms with respect to their own specific inputs: the faults.

Among the various possible levels of application of fault injection (e.g. see [AAACFLMP90]), we used is the physical fault injection technique: in this case, the faults are directly injected on the pins of the integrated circuits (ICs) that implement the prototype of the target system. Although this methodology can only be applied at the final stages of the development process, its main advantages are that the tested

prototype is close to the final system and that it enables a global validation of a complex system integrating both hardware and software issues.

With respect to the validation of the Delta-4 architecture, fault injection can be seen as an implementation testing approach that is actually complementary to the other, more formal, validation methods. In particular, the formal verification work provided the definition of predicates on the fault tolerance mechanisms to be tested by the fault injection experiments and the experimental measures obtained will help in characterizing the global model for dependability evaluation of the complete Delta-4 architecture.

Markov chains were chosen as the modeling technique for **dependability evaluation** of the architecture. The major advantages of using Markov chains are mainly due to: i) their ability to account for stochastic dependencies among the various processes that impact on system behaviour (e.g., error processing, fault treatment and maintenance processes), and ii) the fact that the various dependability measures can be computed easily and efficiently from the same model by means of the existing dependability evaluation methods and tools [Laprie83].

It should be pointed out that it is very difficult to construct directly a global model; so a progressive method is used. First, a global evaluation strategy in terms of inter-connected sub-models is established. After the study of the sub-models, it is necessary to aggregate them and study the global model. This decomposition into sub-models also gives early feedback on the design of the different components included in the sub-models and simplifies the whole dependability evaluation study. Three evaluation objectives were defined based on the three main levels that can be distinguished in the Delta-4 architecture:

- modeling and evaluation of the communication hardware,
- extension of the communication architecture model to include the host-resident management information base,
- establishment of the global model of a target application and evaluation of its dependability.

In this paper we summarize the main results achieved with respect to the first objective.

The paper is composed of three sections corresponding to the main validation activities identified above: each section outlines the features of the validation method which is used and summarizes the major results of the corresponding activity.

2. FORMAL PROTOCOL VERIFICATION

The formal design verification is aimed at detecting inconsistencies in the specifications, and therefore design faults may be removed at an early stage of development. If the specifications of the protocols and of the services to be delivered are provided in a formal way, verification can be a formal process consisting in automatically checking that the specified protocols deliver indeed the specified services.

The formal verification task reported here focus on results obtained for the AMP protocol [BGRRRV90-a] and IRp protocol whose detailed description is presented in [CRPRV91].

2.1 Formal verification using model checking techniques

For design verification in Delta-4, we used model checking techniques as they can be automated to a large extent and are therefore applicable to complex systems. Usually these methods are restricted to finite state systems and thus are suitable for communication protocols.

Formal verification means: "given a *satisfaction relation* between models and service specification described by a set of properties, check whether a pair (model, properties) belongs to this relation.

The model

The system to verify is specified as a set of communicating processes, each process being modeled by a transition system, where states represent the values of variables and transitions correspond to atomic *actions*. These actions may be either internal ones occurring independently of the other processes or communications with the other processes.

The global model is also a transition system representing the behaviour of the whole set of communicating processes. It is obtained as some parallel composition of the transition systems associated with each elementary process. The state space of the global model is a subset of the Cartesian product of the state spaces of the elementary processes. The global transitions correspond either to the interleaving of internal actions of elementary processes, or to communication actions concerning two or more processes (thus, the semantics of communication is by *rendezvous*).

The service specifications

The service specifications define the expected observable behaviour by a set of properties given in the form of formulas of a temporal logic or by automata accepting infinite sequences of actions (Buchi automata).

Two kinds of properties are needed to specify a service: *safety and liveness properties*. *Safety* properties express some invariants remaining always true for the behaviour of the system, as for example deadlock freedom or preservation of message order. *Liveness* properties guarantee some progress in the system, as for example "a process will terminate", "a message sent will eventually reach its recipient".

The verification steps

In our case, verification is carried out in two steps:

- 1) Provision of formal specifications of the service to be delivered and of the protocol machine, and formalization of the assumptions about the behaviour of the environment of the protocol machine. The service is described by a set of properties characterizing execution sequences of the expected behaviour, and the protocol is described by the state machine implemented in the protocol software.
- 2) Generation, from the previous specifications and in accordance with the formal semantics of the specification languages used, of a model of the behaviour of the protocol machine embedded in its environment, and the exhaustive check of the specified service against this model by using appropriate algorithms.

If inconsistencies are detected during the second step, the analysis of the model is used to suggest modifications in the specifications. After correction the specifications are verified again.

The second step and the construction of execution sequences leading to erroneous states are completely automated by using the Xesar tool developed at LGI/IMAG [RRSV87, GRRV89 -a]. Xesar implements model checking techniques for formulas of the temporal logic CTL or for deterministic automata on a model generated from a program written in Estelle/R, a variant of ISO Estelle language, in which communication is defined by rendezvous semantics. The Estelle/R formal semantics is based on transition models with explicit clocktick events.

In Xesar, model checking algorithms, requiring explicit representation of the model, are implemented for safety and liveness properties expressed by CTL formulas. Xesar provides also algorithms to efficiently verify safety properties given by a particular class of Buchi automata on even quite large models.

2.2 Verification of protocols in Delta-4

Practically, a finite set of judiciously selected configurations of fixed number of networked protocol machines is exhaustively verified.

The formal verification allows analysis of specific configurations of the execution of a protocol. In each tested configuration, exhaustive analysis allows detection of all design errors, even those occurring in complex execution sequences and which are very unlikely to be detected by simulation or test, as they have very low probability.

To infer the validity of the protocol in any configuration, one needs some "inductive" method. Different solutions have been proposed, generally based on an invariant characterizing the behaviour of the protocol. This invariant must have a "good" structure, that is, it must allow induction on the structure of the configuration. However, the protocols considered are more complex, as they are based on the (simultaneous and successive) use of different paradigms (two phase protocols, election, reliable multicast, use of time out to avoid livelocks, ...). The complexity of the interaction of these paradigms makes hard the description of the complete behaviour of the protocol by a "suitable" invariant.

But for any particular protocol, informal considerations, based on the overall structure, provide convincing arguments that the general validity can be deduced from the verification of a limited family of configurations. For example, it can be argued that in the case of AMP, configurations with three machines are sufficient, and that we can further limit the possible actions in any of these machines.

Representation of the environment

A crucial point in the verification is the choice of an adequate model, suitable in the present case to time bounded systems, such that all properties formally verified on the model must be true in reality. This adequacy depends deeply on the abstraction level used for the protocol machine description and on the modeling of its environment. For example, if the environment is modeled by a process "chaos" (i.e., that can modify transmitted messages in any possible way), all detected errors may not correspond to real errors in the protocol.

The environment consists of adjacent layers in the communication stack; the model must take into account the different features characterizing the behaviour of these layers, such as buffering, occurrence of faults, and timing constraints.

The first two features are explicitly represented in the model. Concerning time, two kinds of modeling have been used, depending on the nature of the system to be verified:

- in systems where all the message transmissions are implicitly clocked, as for example by a token circulation mechanism as in the Turbo MAC AMp protocol, timeless models are used [GRRV89-b]; the verification of service properties is made under some fairness assumptions, e.g. stating that the property is true provided the token is effectively circulating,
- in other systems, as for example the generic AMp [BGRRRVV90-a] or the IRp [CRPRV91] protocols, the passage of time is explicitly represented in the model by means of a specific *clocktick* action which occurs simultaneously in all processes whereas all other actions are interleaved or synchronised via rendezvous communication; in this case, some statements concerning time limits can be verified, but the results depend on the particular values of timers taken into account in the specifications.

Abstraction of the protocol machine

In the specifications of the protocol machines, we have only defined the representations of the data structure relevant for the verification. It is worth noting, however, that all transitions were actually fully specified.

2.3 The major results of the verification

The starting material for the formal verification activity was an already-existing communication stack software that had been partially and informally specified. The results obtained required tight interaction between the designer-implementer and the specifier-verifier teams: they mainly concerned the formal specifications from which the new implementations have been structured and developed, the clarification of the assumptions made on the behaviour of the environment, and the formal verification that increases the confidence in the correctness of the protocol designs.

Two families of protocols have been verified, and consistent structured specifications of both protocol machines and services have been provided:

- reliable group or multicast communication protocols, (Generic AMP, Turbo MAC AMp) with a strong quality of service (atomic multicast) characterized by a set of properties given in the table of figure 1 (the assumptions about the behaviour of the network, a part of the environment, have been formalized in a similar way [G90050; chapter 10],
- inter replicate coordination protocols (IRp) managing the replicated aspects of the session service users, and supporting distributed fault tolerance.

Concerning the table of figure 1, it is worth noting that these properties characterize what is meant by a *proper* service, and can thus be used as a basis for verifying the protocol design and its implementation. For the formal verification aim, this set of properties is translated into a suitable formalism.

Particular configurations with a fixed number (e.g. 3 for the generic AMp) of interconnected protocol machines, covering critical cases derived from the structure of the protocol, have been verified with the Xesar tool. The generated models have reasonable size (on average 350,000 states). Non trivial inconsistencies that could not

be obtained with classical simulation or testing techniques, have been detected (and corrected) in the specifications.

Unanimity: any message* delivered to a correct participant** is delivered to all correct participants.
Non-triviality: any message delivered was sent by a correct participant.
Accessibility: any message delivered was delivered to a participant correct and accessible for that message.
Delivery: any message is delivered, unless the sender fails or some participant(s) is(are) inaccessible***.
Consistent causal order: any two messages delivered to a participant are delivered to all correct participants, in the same order which obeys causality.
Termination: messages are delivered within a known bounded time.
Consistent Group View: given any receive ordering by the participants of the group, each change to group membership is indicated, in a consistent order, to all correct group participants.

* A message is an AMP service data unit.

** A correct participant is a participant residing on a fault-free station.

*** The accessibility concept is related with recipient constraints such as lack of resources, which may cause it to sporadically refuse to accept a messages. The recipient, however, signals its refusal to the sender, allowing the protocol to progress and terminate.

Figure 1 - Set of properties defining the AMP service

Two types of inconsistencies have been identified: "surface" ones which can be easily corrected, and "deep" ones which need more analysis to be corrected; in fact they may show up inconsistencies in the application of the paradigms underlying the design of the protocol.

A first example is given by the generic AMP, where significative "surface" inconsistencies are detected such as bad initialisation of local variables, wrong actual parameters, too weak conditions in transitions. A case of "deep" inconsistency corresponding to a livelock situation is the following: in some global states a "monitor election" procedure can be restarted forever and will thus never terminate. The formal specifications and the results of verification of the generic AMP protocol are discussed with more details in [BGRRRV90-b].

A second example is given by the IRp protocols. Although the verification is not yet completed, some abnormal situations have already been detected. The IRp protocols ensure that a family of replicate entities perform the same actions, and that an external user perceives all the replicates as a single one. Each correct replicate has a complete knowledge of the state of the other correct replicates. To enforce this point, a replicate uses an AMP protocol (atomic multicast) in order to inform the other replicates of any internal change. When a message is sent outside of the family of replicates, such a method is not appropriate, as only one copy of the message must be sent. Election is used to select exactly one sender. Timers are used to control the relative speed of execution of the different replicates.

These paradigms (atomic multicast, election, timeouts) are used together. As the number of possible behaviours is very large, conflicts occur. For example, two inconsistencies have been detected in the message send protocol:

- After one replicate has been elected as a sender, if it dies, the other replicates are sometimes not able to elect another sender.
- If timers expire exactly at the time of the election of a sender, due to transmission delays, it can happen that the sender sends the message, but also that a new election is performed, and therefore the message is sent twice.

This verification work allowed the detection of a large number of errors in the specifications of the protocol software.

3. IMPLEMENTATION VALIDATION BY FAULT INJECTION

The experimental validation summarized in this section is based on the use of physical fault injection for the validation of an 802.5 token-ring Delta-4 architecture.

3.1. Validation objectives

Two important contributions of fault injection concern the verification of the fault-tolerance mechanisms and the characterization of their behaviour, thus enabling weakness in their design and/or implementation to be revealed. Also, a statistical analysis of the responses obtained during the fault injection experiments enables some relevant dependability parameters - coverage, fault dormancy, error latency, etc. - to be estimated. More specifically, the experiments carried out were aimed at:

- i) **estimating two levels of coverage** provided by the fault tolerance mechanisms: a) the local coverage (**C_D**) achieved by the detection mechanisms which control the extraction of the NACS, b) the distributed coverage (**C_T**) corresponding to the **fault tolerance** provided by the defensive characteristics of the AMP software,
- ii) **testing**, in the presence of faults, the **service provided by AMP**.

It is worth mentioning here that the AMP tested had not been previously submitted to the formal verification described in the previous section.

3.2 Characterization of the fault injection test sequences

The target system is made up of four stations S_i , $i = 1, \dots, 4$ (host + NAC) interconnected by a token-ring Delta-4 network.

A general distributed testbed has been developed that enables the automatic control of the fault injection experiments [MACFP90]. Faults were injected within the NAC of station S1, using the fault injection tool MESSALINE [ACL89].

The table of figure 2 summarizes the major characteristics of the test sequences.

NAC Component	AMP Release	Faults Injected	# IC's Tested	# Experiments
type 1 NAC	V 1	Intermittent	40	4799
		Transient	3	600
		Permanent	3	750
	V 2	Intermittent	8	1200
	V 2.3	Intermittent	8	1200
type 2 NAC	V 2.5	Intermittent	21	3150
Total	—	—	61 + 24	7949 + 3750

Figure 2 - Organization of the test sequences

Two distinct hardware architectures of the target system have been tested:

- i) an **homogeneous architecture** where each station was made up of a Bull SPS7 host interconnected by a **type 1 NAC** based only on restricted self-checking mechanisms,

- ii) an **heterogeneous architecture** where the injected station is made up of a Ferranti Argus 2000 host computer associated with a **type 2 NAC** featuring improved (duplex) self-checking mechanisms; three Bull SPS7/type 1 NAC stations complement the system.

The corrections induced by the design/implementation errors unveiled in the preliminary experimental validation phase carried out on an AMP version not previously submitted to formal verification, led to the elaboration of several **releases of AMP software**. The successive fault injection test sequences can thus be regarded as a kind of *regression testing* effort.

So as to account for the most likely faults, the injected faults were mainly **intermittent** faults, but included also transient and permanent faults. Faults were injected by forcing voltage levels on single or multiple pins of the ICs implementing the injected NAC component. Even though only a subset of ICs were tested, the use of the **forcing** fault injection technique means that a high proportion of actual equipotential lines on the boards are faulted. In particular, for the type 1 NAC, the resulting pin coverage was more than 84 %.

The total row sums up the ICs tested and the fault injection experiments. The 7949 experiments on 61 (40 + 21) ICs form the experimental results used for the comparison of the two NAC architectures. The other 3750 experiments on 24 ICs correspond to selective experiments carried out on the type 1 NAC to analyse i) the impact of the duration of the injected faults (1350 experiments on 2x3 ICs) and ii) the V 1 and V 2.3 releases of the AMP (2400 experiments on 2x8 ICs).

Beside the traces and memory dumps used for AMP debugging, the readouts collected to assess the efficacy of the fault tolerance consisted of:

- i) the activation of the injected fault as an error (predicate **E**) and the related instant **T_E**,
- ii) the status of the insertion relays (predicate **I_i** = true, if the NAC of station *i* is inserted in the ring and the associated timing characteristics (**T_{D_i}**),
- iii) The test of the conjunction of the major AMP properties (predicate **P**), e.g. *order*, *unanimity*, etc.; see figure 1 and [G90050; chapter 10] for a definition of the AMP properties.

The measures considered in the analysis consist of two types of measures: **predicates** and **time distributions**. Predicate $D = I \wedge E$ defines the error detection predicate and thus, the coverage **C_D** characterizing the efficiency of the NAC detection mechanisms is estimated by:

$C_D \approx \frac{|D|}{|E|}$ Predicate **C** = **I₂ ∧ I₃ ∧ I₄**, characterizes the confinement of the fault/error (i.e., all non-injected stations remain inserted). Thus, **T** = **P ∧ C ∧ E**, defines the fault tolerance predicate and the coverage **C_T** that globally assesses the defensive properties of the protocol at the MAC layer, can then be expressed as: $C_T \approx \frac{|T|}{|E|}$.

If **T_F** denotes the instant of fault injection, then, **T_d** = **T_E - T_F** measures the **fault dormancy** interval and **T_i** = **T_{D1} - T_E** measures the **extraction latency** that corre-

2 The notation $|X|$ designates the cumulated number of assertions of predicate *X* over the set of experiments.

sponds to the time interval between an **error** and the **extraction** of the NAC of the injected station.

3.3 Experimental results

We focus here on the reliability growth resulting from the debugging of the AMP software and on the comparison of the type 1 and type 2 NAC architectures.

The complete results can be found in [AACFMP89-a], [AACFMP89-b], [AACFMP90] and [ACMP1].

Figure 3 shows the distribution of the error codes observed when a fault was not properly tolerated by the protocol, i.e. when predicate C was not true) among the four major functions of the AMP software: monitor, emitter, receiver and driver.

Each ring indicates in its centre the total number of error codes collected and shows the relative distribution of these error codes among the modules.

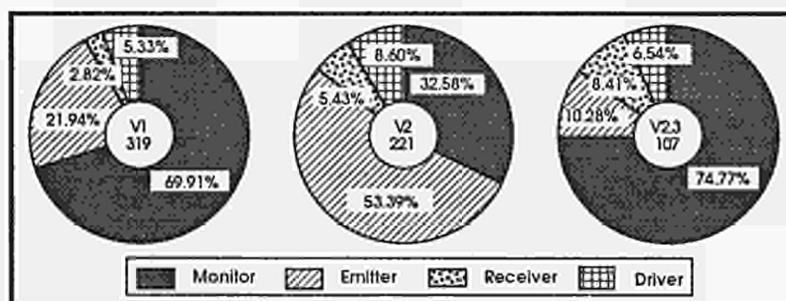


Figure 3 - Synthesis of the error codes per software function

These figures indicate that the number of errors has been continuously decreased (it has been divided by three between V1 and V2.3). Due to the distribution observed for V1, the main debugging effort was placed on the monitor module. This resulted in release V2 for which it was observed an increase of the proportion of error codes concerning the emitter module; it is likely that the observation of these errors was masked by the large proportion of errors affecting the monitor module during the test of V1. Release V2.3 enabled these errors to be significantly reduced while maintaining about the same number of errors related to the monitor module.

Figure 4 shows the significant impact of the debugging of AMP and of the improvement of the self-checking mechanisms of the NAC on C_T . The results concerning the type 2 NAC gather only those obtained for the first 10 circuits tested. Although it is based on a partial analysis of the experimental results³, it is worth noting that this estimation can be regarded as a pessimistic estimation due to the specific functions played by these circuits (see [ACMP91]).

These results show that almost half of the 20% increase observed on C_T —actually 9% is due to the last hardware/software combination tested.

³ The notation $\sum X$ designates the cumulated number of assertions of predicate X over the set of experiments.

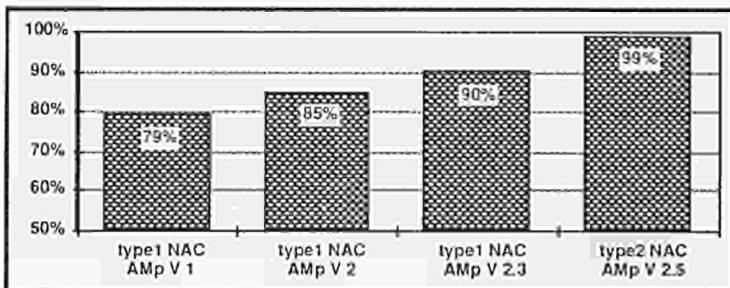


Figure 4 - Evolution of the distributed coverage

Figure 5 plots the cumulative distributions of the detection latency observed for the type 1 NAC (V2.3) and the type 2 NAC (V2.5).

The curves show that, while only 30 % of the errors are detected in less than 1 s for the type 1 NAC, the corresponding percentage of errors detected is greater than 80 % for the type 2 NAC. Furthermore, the asymptotic levels (obtained for 100 s) identify the respective measures of the local coverage C_D .

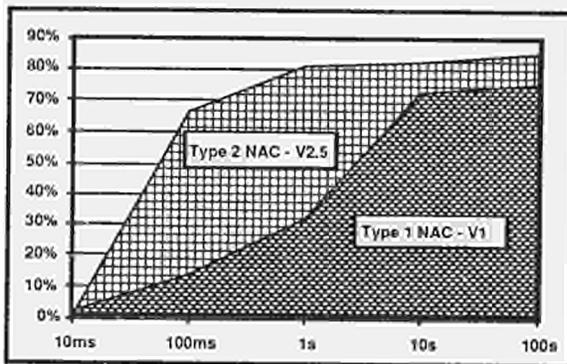


Figure 5 - Cumulative latency distributions for the type 1 and type 2 NACs

Although these results correspond to distinct releases, it should be pointed out that no noticeable modification was observed for the type 1 NAC between the results obtained for release V2.3 and the (partial) tests carried out for release V2.5. This supports the conclusion that the significant improvement noted in the latency between the type 1 NAC and the type 2 NAC is mainly a consequence of the improvement of the self-checking capabilities implemented in the type 2 NAC architecture.

4. DEPENDABILITY EVALUATION OF THE COMMUNICATION SYSTEM

The various hardware communication architectures considered in Delta-4 are the 802.4 token bus, the 802.5 and FDDI token rings. Each of them can use a single (non redundant) medium or duplicated media.

The set of the NACs with the underlying media constitute the communication system that is evaluated in this section. There are thus two essential aspects to be taken into account in the models: the communication topology and the nature of the NACS, i.e. type 1, and type 2.

4.1 Model assumptions

For modeling purposes, we will assume that the failure rates associated to the different components of the hardware structure are constant. This assumption allows a Markov modeling approach to be used. This assumption is realistic for accidental events such as physical failures and it has been shown that it is a good assumption for maintenance processes if the mean maintenance duration is small compared to the mean time to failure [Laprie75].

Dependability can be evaluated from the Markov chain either directly - when the models are not very complex - or using a dependability evaluation tool such as the SURF program [LCL81] when the models are more complex.

Tools do not usually give analytical expressions of the different measures due to the complexity of the models. However, when the non failed states of a Markov chain constitute an irreducible set (i.e. the graph associated with the non absorbing states is strongly connected), it can be shown that the absorption process is asymptotically a homogeneous Poisson process; approximate expressions of the measures of dependability can be obtained through the "equivalent" failure rate [PG80].

The aim of this technique is to transform the initial Markov chain to a reduced Markov chain made up of two states: the non failed-state and the failed state; the asymptotic transition rate, λ_{eq} , from the non failed-state to the failed state is called the equivalent failure rate. This transition rate is obtained directly from the initial chain and can be expressed as a function of the failure and repair rates of the different components of the system [PG80].

Reliability, $R(t)$, and the asymptotic unavailability, UA , are then given by:

$$R(t) = \exp(-\lambda_{eq}t), \text{ and } UA = \frac{\lambda_{eq}}{\mu},$$

where μ is the repair rate from the failed state.

Since the Delta-4 system is intended for applications in which repair is possible, the associated chains are generally strongly connected so this approach was adopted: the different communication systems are evaluated through their equivalent failure rates.

We assume that a non-covered failure of one element leads to a total system failure (which is a pessimistic assumption). Concerning the maintenance policy, we assume the following:

- a covered failure (of the NAC or of the medium) does not affect service delivery, moreover the repair of such a failure does not need service interruption,
- after a non-covered failure, (of the NAC or of the medium), service delivery is interrupted and repair of all the failed elements is carried out before service is resumed,
- in case of one or several covered NAC failures, followed by a covered failure of the medium, repair priority is given to the medium,
- for the dual ring, the wiring concentrators have the highest repair priority.

4.2 Notation and numerical values of the parameters

The two types of NACs are modeled in the same manner, they differ by the numerical values of the parameters: for the type 2 NAC, error detection coverage should be higher, the failure rate is also higher due to the greater amount of hardware necessary to

enhance the self-checking. The dual FDDI ring has the same model as the dual 802.5 ring.

The main parameters of the models are:

- λ_N : the failure rate of the NAC - a value of 10^{-4} / h has been taken as a reference and corresponds to 1 failure per year,
- λ_{wc} : the failure rate of a wiring concentrator in the dual ring, it should be about the same as the failure rate of a NAC (it has been taken in fact equal to λ_N in this study),
- n : the number of stations - fixed (arbitrarily) at 15,
- N : the number of wiring concentrators in the dual ring,
- λ_B : the failure rate of the bus - a value of $2 \cdot 10^{-5}$ / h has been taken; this corresponds to 1 failure per 5 years,
- λ_R : the failure rate of the ring - it has been taken equal to λ_B ,
- μ : the repair rate, a repair duration of 2 hours has been adopted.

The coverage factors for the different elements are denoted: p_N , for the NAC, p_B , for the bus, p_L , for a link in the ring, p_{wc} , for the wiring concentrator and p'_{wc} , for the correct reconfiguration of the dual ring after a non-covered failure of a wiring concentrator.

Let \bar{p}_X be defined as: $\bar{p}_X = 1 - p_X$ where $X \in \{N, B, L, WC\}$

4.3 Summary of the results

With these notations the equivalent failure rates of the different architectures are very complex for the dual media. However they can be simplified using the fact that:

$$\lambda_B/\mu \ll 1, \lambda_R/\mu \ll 1 \text{ and } \lambda_N/\mu \ll 1.$$

The expressions of these failure rates considering only the first order terms are:

$$\text{Single ring: } \lambda_{eq} \approx \lambda_R + n \bar{p}_N \lambda_N + n \cdot p_N \cdot \frac{\lambda_N}{\mu} \cdot [\lambda_R + (n-1) \bar{p}_N \cdot \lambda_N] \lambda_B + (n-1) \bar{p}_N \cdot \lambda_N$$

$$\text{Dual ring: } \lambda_{eq} \approx 2 \bar{p}_L \lambda_R + n \bar{p}_N \lambda_N + N \bar{p}_{wc} p'_{wc} \lambda_{wc}$$

$$\text{Single bus: } \lambda_{eq} \approx \lambda_B + n \bar{p}_N \lambda_N + n \cdot p_N \cdot \frac{\lambda_N}{\mu} \cdot [\lambda_B + (n-1) \bar{p}_N \cdot \lambda_N]$$

Dual bus:

$$\lambda_{eq} \approx 2 \bar{p}_B \lambda_B + n \bar{p}_N \lambda_N + 2 n \frac{\lambda_N}{\mu} \lambda_B [p_N \bar{p}_B + p_B \bar{p}_N] + n (n-1) p_N \cdot \bar{p}_N \frac{\lambda_N}{\mu} \lambda_N$$

It can be noted that, for the single media the equivalent failures rates are limited by the failure rate of the medium and the non-covered failures of the NACs and that, for the dual media, they are directly related to the failure rate of the non-covered failures. The coverage factors are thus of prime importance. However, due the numerical values of the different failure rates the coverage of the NACs, p_N has more influence than p_B and p_L .

With the value taken for the mean repair time ($1/\mu$) (2 hours, which is relatively low), the dominant source of unavailability is lack of coverage rather than exhaustion of

redundancy; i.e. the initial terms in the equivalent failure rate and in the unavailability expressions dominate; the unavailability is therefore directly proportional to the mean repair time ($1/\mu$).

For the ring, duplication is worthwhile only for $\lambda_R > 5 \cdot 10^{-5}/h$ even with a perfect coverage of the NACs, $p_N = 1$, as shown on figure 6. This is due to the introduction of the wiring concentrators whose failures rates are of the same order of magnitude as the NACs.

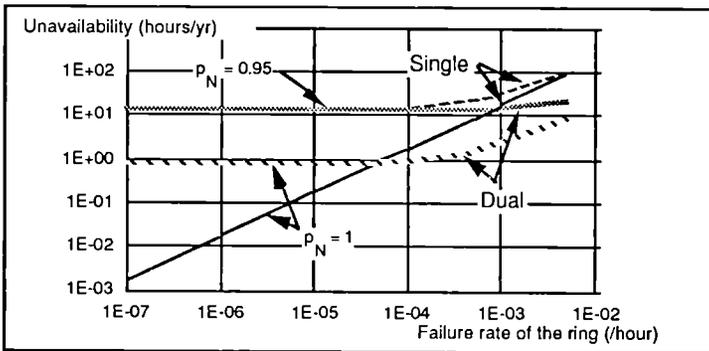


Figure 6 - Communication system unavailability for the single and the dual ring ($P_L = 0.95$ and $PWC = P'WC = 0.9$)

For instance, considering $p_N = 1$, duplication of the ring acts as follows:

- for $\lambda_B = 10^{-5}/h$, it increases the unavailability from 11 mn to 48 mn per year,
- for $\lambda_B = 10^{-4}/h$, the unavailability is raised from 1 h 45 mn to 58 mn per year.

With respect to dependability improvement due to the duplication of the bus, the unavailability of the communication system with a single and a dual bus, versus the failure rate of the bus (λ_B) and for p_N 0.95 and 1, is given in figure 7.

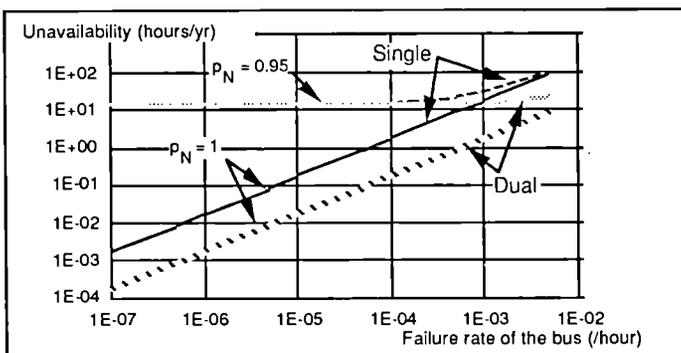


Figure 7 - Communication system unavailability for the single and the dual bus

When the coverage factors are less than 1, duplication can lead to dependability deterioration depending on the value of the bus failure rate: for instance, for $p_N = p_L$

= 0.95, improvement is effective only for $\lambda_B \geq 2 \cdot 10^{-5}/h$. This is due to the fact that when λ_B is low, dependability is conditioned by the NAC failures.

When the coverage factor of the NAC is 1, duplication is worthwhile, for example:

- for $\lambda_B = 10^{-5}/h$, duplication of the bus decreases the unavailability from 11 mn per year to 1 mn per year,
- for $\lambda_B = 10^{-4}/h$, unavailability is decreased from 1 h 45 mn to 11 mn per year, which is a significant improvement.

4.4 Communication systems comparison

Concerning comparison of these architectures, it is very difficult to classify them. Assuming the same failure rate for the bus and for the ring leads to the same expression of the equivalent failure rate and for the unavailability of the single medium. Figures 5 and 6 show that for reasonable failure rates ($\lambda_B \approx \lambda_R < \lambda_N$) dependability measures are independent of this failure rate. This means that the single bus and the single ring are equivalent.

In the case the ring, duplication of the medium can actually deteriorate the dependability measures depending in the parameter values. The results enable the different architectures to be compared according to the various parameters in order to make a tradeoff and to select the most suitable architecture. For instance, for the considered values, the dual bus seems more interesting than the dual ring for $\lambda_B < 4 \cdot 10^{-3}/h$; however, the value of the failure rate of the wiring concentrator is of prime importance: a lower value of λ_{WC} (e.g. passive WCs) acts in favour of the dual ring.

4.5 Software reliability

Quantitative assessment of software reliability is usually carried out through the application of reliability growth models. These models enable prediction of either the number of failures to be activated for the next period of time or the mean time for the next failures or the software failure rate or some combinations of these measures.

Reliability growth models are generally parametric models and these parameters have to be estimated (i.e. model calibration) to carry out dependability predictions. Calibration of the models is enabled via failure data collection on the software either during development or in operation. Predictions are thus based on:

- observation of the software behaviour during a given period of time,
- calibration of the model using the observed failure data,
- application of the model to estimate dependability measures.

Reliability growth models will be applied when sufficient failure data items have been collected. As the profile of the software changes, it is not possible to determine in advance which model will be best fit the data. Therefore the data will be input to the reliability growth models available to SRD and LAAS and the results will be analysed to ascertain the best model for the software in the same manner as in [KS 87, KBM 91].

Designing, establishing and implementing a data collection facility which is effective but also uses cost-effective techniques is a long process. However, data collection has been carried out on the developed software, not only the Delta-4 Application Support Environment but also the Delta-4 Communication software (MCS) and some of the results are now being processed for input to the reliability growth models.

To complement the collection of failure data, static analysis of the new version of the code is also being conducted as soon as each new version is released.

5. CONCLUSION

In order to face the complexity of the task, the validation process involved the use of three complementary validation approaches, namely: formal verification, experimental validation by means of fault injection and dependability evaluation based on the use of Markov chains.

Furthermore, due to the unrealism of performing a *global* validation of the whole Delta-4 architecture, the validation work actually carried out has focused on the more innovative and tringent aspects of the Delta-4 architecture: the communication protocols and the self-checking mechanisms. In spite of this - unavoidable - partial detailed coverage, this work has proved very fruitful in the actual impact it had on the project development.

The **formal verification** approach had a very significant influence in the structuring of the protocols. More precisely, the major benefit of the formal verification work is that it enforces the developers to provide formal abstract specifications of innovative protocols, consisting of three parts:

- environment assumptions,
- protocol state machines with a suitable structure,
- service to be delivered characterised by a set of well-identified properties.

As these specifications are formally verified, they are consistent. It is also worth noting that they facilitated argumentation about the protocol software and the other investigations - in particular the experimental validation - of the protocol itself.

The **experimental validation** work has actually exemplified the two facets of fault injection within the validation process (fault removal and fault forecasting).

As an integration testing including interactions between hardware and software components, the fault injection experiments did help in improving the protocol software and also in rating the behavior of successive releases in presence of faults. These experiments also proved useful for estimating the coverage provided by the error detection and recovery mechanisms of the specific fault tolerance mechanisms imbedded in the network attachment controllers and in the reliable communication protocols.

Concerning the efficiency of the self-checking mechanisms of the network attachment controllers, it is worth noting that fault injection actually evidenced the benefits obtained from the use of improved fully redundant self-checking mechanisms. Furthermore, all these experimental measures provide sound references upon which to base numerical dependability evaluation of the global service provided by the Delta-4 architecture.

Concerning **dependability evaluation**, we have derived basic communication models with four possible communication topologies: single and dual bus, and single and dual ring. The main results concern the derivation of analytical expressions of the failure rate and unavailability of the various communication systems. It is shown that, for the single media configurations, the equivalent failures rates are limited by the failure rate of the medium and non-covered failures of the NACs and that, for the dual media configurations, they are directly related to the failure rate of the non-covered failures only.

These models are parametric; parameter values have been chosen so as to form a coherent set in order to compare the different architectures and to obtain a first estimate of the measures of dependability. Evaluation enabled identification of the most critical

parameters and showed that it is not possible to draw any definite conclusion as to a dependability ordering of the various topologies since the results are closely related to the values of these critical parameters. The estimated values need to be replaced by "real" values issued from field data, either by direct evaluation or by measurement. In particular, fault-injection can be used for measuring coverage factors as stated in section 3. However, it is shown that - whatever the architecture - the coverage factor of the NAC is of prime importance; it is thus worthwhile to put emphasis on this coverage (i.e. self-checking mechanisms) during development and on the measurement of the coverage level actually obtained.

All these benefits were the results of a significant investment in the project (money, time and people) working together on the validation issue. These gains necessitated a large number of interactions between the partners of the consortium (constructors, designers, modelers, testers, etc.). Indeed, this was made possible by the actual synergism that took place within the project. This constitutes another kind of fully rewarding outcome rather difficult to rate but of very significant importance from the points of view of transfer of expertise, modification of technical development process and identification of practical problems to tackle.

ACKNOWLEDGEMENT

The technical support and the feedback received from **Bull** and **Ferranti** was of great value during the whole validation process. In particular, for setting up the fault injection test sequence and for refining the analysis of the experimental results, we would like to thank Marc Chèreque and Rendé Ribot (Bull) and Nigel Howard (Ferranti).

Our thanks also go to Jean-Charles Fabre (member of the TSF group at **LAAS**) for his significant contribution in updating the testbed software and to Martine Aguéra (from the "Informatics and Instrumentation" Department at **LAAS**) for her help in setting up and maintaining the experimental testbed. We would also like to thank Carlos Rodriguez (Verilog) whose contribution has been determinant in the Generic AMP verification.

REFERENCES

- [**AAACFLMP90**] J. Arlat, M. Aguera, L. Amat, Y. Crouzet, J.-C. Fabre, J.-C. Laprie, E. Martins, D. Powell, "Fault-Injection for Dependability Validation - A Methodology and Some Applications", *IEEE Trans. on Software Engineering, Special Section on Experimental Computer Science*, vol. 16, February 1990, pp. 166-182.
- [**AACFMP89-a**] M. Aguera, J. Arlat, Y. Crouzet, J.-C. Fabre, E. Martins, D. Powell, "Results of Fault-Injection into an MCS Network Attachment Controller with Limited Self-Checking", *LAAS Research Report 89-071, Delta-4 Document n° R89.022*, March 1989, Revision May 89, 68 p..
- [**AACFMP89-b**] J. Arlat, M. Aguera, Y. Crouzet, J.-C. Fabre, E. Martins, D. Powell, "Dependability Testing Report LA1", *LAAS Research Report 89-410, Delta-4 Document n° R89.139*, December 1989, 35p..
- [**AACFMP90**] J. Arlat, M. Aguera, Y. Crouzet, J.-C. Fabre, E. Martins, D. Powell, "Experimental Evaluation of the Fault Tolerance of an Atomic Multicast Protocol", *IEEE Trans. Reliability, Special Issue on Experimental Evaluation of Computer Reliability*, vol. 39, n° 4, October 1990, pp. 455-467.
- [**ACL89**] J. Arlat, Y. Crouzet, J.-C. Laprie, "Fault-Injection for Dependability Validation of Fault-Tolerant Computing Systems", in *Proc. 19th IEEE Int. Symp. Fault-Tolerant Computing*, Chicago, Illinois, USA, June 1989, pp. 348-355.

- [ACMP91]** J. Arlat, Y. Crouzet, E. Martins, D. Powell, "Dependability Testing Report LA2 Fault Injection on the Fail-Silent NAC: Preliminary Results", *LAAS Research Report 91-043, Delta4 Document n° R90.206*, March 1991, 47p..
- [BGRRRVV90-a]** M. Baptista, S. Graf, J-L. Richier, L. Rodrigues, C. Rodriguez, P. Verissimo, J. Voiron. "Formal Specification and Verification of a Network Independent Atomic Multicast Protocol", in J. Quemada, J. Mañas, and E. Vazquez, editors, *3th International Conference on Formal Description Techniques (FORTE'90)*, North-Holland, 1990, pp. 425-434.
- [BGRRRVV90-b]** M. Baptista, S. Graf, J-L. Richier, L. Rodrigues, C. Rodriguez, P. Verissimo, J. Voiron. "Generic AMP Verification report", *Delta4 Document n° R90.225*, December 1990.
- [CB89]** R. Chillarege, N.S. Bowen, "Understanding Large System Failures - A Fault Injection Experiment", in *Proc. 19th IEEE Int. Symp. Fault-Tolerant Computing*, Chicago, Illinois, USA, June 1989, pp. 356-363.
- [CES83]** E. M. Clarke, E. A. Emerson, E. Sista, "Automatic Verification of Finite State Concurrent Systems using Temporal Logic Specifications: A practical Approach", in *Proc. 10th Symp. on Principles of Programming Languages (POPL 83)*, ACM, 1983. Complete version published in *ACM TOPLAS*, 8(2), April 1986, pp. 244-263.
- [CRPRV91]** M. Chèreque, P. Reynier, D. Powell, J.-L. Richier, J. Voiron, "Coordination of Active Replicates within Delta4 Open Systems Architecture: the Inter Replicates Protocol (IRp)", submitted to ECW 9 1.
- [CS90]** E.W. Czeck, D.P. Siewiorek, "Effect of Transient Gate-Level Faults on Program Behavior", in *Proc. 20th IEEE Int. Symp. Fault-Tolerant Computing*, Newcastle upon Tyne, Angleterre, June 1990, pp. 236-243.
- [FRV86]** J-C. Fernandez, J-L. Richier, J. Voiron, "Verification of Protocol Specifications using the Xesar system", in *Protocol Specification, Testing, and Verification, V*, North-Holland, 1986, (*Proc. IFIP WG 6.1 5th Int. Conf. on Protocol Specification, Testing and Verification, Moissac, 1985*), pp. 71-90.
- [G90050]** Delta4 Architecture Guide, The Delta-4 Project Consortium, (Editor: D. Powell), *Delta Document n° G90.050*, December 1990.
- [GKT89]** U. Gunneflo, J. Karlsson, J. Torin, "Evaluation of Error Detection Schemes Using Fault Injection by Heavy-ion Radiation", *Proc. 9th IEEE Int. Symp. Fault-Tolerant Computing*, Chicago, Illinois, USA, June 1989, pp. 340-347.
- [GRRV89-a]** S. Graf, J-L. Richier, C. Rodriguez, J. Voiron, "What are the Limits of Model Checking Methods for the Verification of Real Life Protocols?" In *Workshop on Automatic Verification Methods for Finite State Systems*, Grenoble, LNCS 407, Springer Verlag, 1989, pp. 275-285.
- [GRRV89-b]** S. Graf, J-L. Richier, C. Rodriguez, J. Voiron, "Protocol Verification Report 1989", *Delta-4 Document n° R89.147*, December 1989.
- [IS089]** ISO. *IS ISOIOSI 9074 - ESTELLE: A formal description technique based on an extended state transition model*. International Standard, ISO, 1989.
- [KBM91]** K. Kanoun, M.R. Bastos, J. Moreira de Souza, "A Method for Software Reliability Analysis and Prediction: Application to the TROPICO-R Switching System", *IEEE Trans. on Software Engineering*, vol. 17, no 4, April 1991.
- [KP91]** K. Kanoun, D. Powell, "Evaluation of Bus and Ring Communication Topologies for the Delta-4 Distributed Fault-Tolerant Architecture", *LAAS Research Report 91-013*, to appear in *Proc. 10th Symp. on Reliable Distributed Systems (SRDSIO)*, September-October 1991, Pisa, Italy
- [KS871]** K. Kanoun, T. Sabourin, "Software Dependability of a Telephone Switching System", in *Proc. 7th IEEE Int. Symp. on Fault Tolerant Computing*, Pittsburgh, USA, June 1987, pp.236-241.

- [Laprie75] J.C. Laprie, "Reliability and Availability of Repairable Systems", in *Proc. 5th IEEE Int. Symp. on Fault-Tolerant Computing*, Paris, France, June 1975, pp.87-92.
- [Laprie83] J.-C. Laprie, "Trustable Evaluation of Computer Systems Dependability", in *Proc. Int. Workshop Applied Mathematics and Performance|Reliability Models of Computer|Communication Systems*, Pisa, Italy, September 1983, pp. 341-360.
- [Laprie89] J.-C. Laprie, "Dependability: A Unifying Concept for Reliable Computing and FaultTolerance", in *Dependability of Resilient Computers*, (Ed. T. Anderson), BSP Professional Books, 1989, Chapter 1, pp. 1-28
- [LCL81] J.C. Laprie, A. Costes, C. Landraut, "Parametric Analysis of 2-unit Redundant Computer Systems with Corrective and Preventive Maintenance", *IEEE Trans. on Reliability*, R-30, June 1981, pp. 139-144.
- [MACFP90] E. Martins, J. Arlat, Y. Crouzet, J.-C. Fabre, D. Powell, "Testing Multiprocessor Protocols in the Presence of Faults", in *Protocol Test Systems*, North-Holland, 1990, (*Proc. IFIP TC6 2nd Workshop on Protocol Test Systems*), Berlin, Germany, October 1989, pp. 77-91.
- [PG80] A.Pagès, M.Gondran, "*System Reliability*", Eyrolles, Paris, 1980, in French.
- [RRSV87] J-L. Richier, C. Rodriguez, J. Sifakis, J. Voiron, "XESAR: A tool for Protocol Validation - User Manual - 1.2 Edition", Laboratoire de Génie Informatique, Grenoble, France, September 1987.
- [SM83] R.L. Schwartz, P.M. Melliar-Smith, "Specifying and Verifying Ultra-Reliability and FaultTolerance Properties", in *Proc. COMPCON'83*, San Francisco, CA, USA, February-March 1983, pp. 71-76.
- [Walter90] C J. Walter, "Evaluation and Design of an Ultra-Reliable Distributed Architecture for Fault Tolerance", *IEEE Trans. Reliability, Special Issue on Experimental Evaluation of Computer Reliability*, vol. 39, n° 4, October 1990, pp. 492-499.

COOPERATION IN INDUSTRIAL SYSTEMS

N.R. Jennings
Dept. Electronic Engineering
Queen Mary and Westfield College
Mile End Road
London E14NS
n-jennings@eurokom.ie
tel: +44-71-975-5358
fax: +44-81-981-0259

Summary

ARCHON is an ongoing ESPRIT II project (P-2256) which is approximately half way through its five year duration. It is concerned with defining and applying techniques from the area of Distributed Artificial Intelligence to the development of real-size industrial applications. Such techniques enable multiple problem solvers (e.g. expert systems, databases and conventional numerical software systems) to communicate and cooperate with each other to improve both their individual problem solving behaviour and the behaviour of the community as a whole. This paper outlines the niche of ARCHON in the Distributed AI world and provides an overview of the philosophy and architecture of our approach the essence of which is to be both general (applicable to the domain of industrial process control) and powerful enough to handle real-world problems.

1. Introduction

After more than a decade of successful exploitation there are now over 100,000 expert systems being used in hundreds of companies all over the world to solve complex problems in numerous domains [10]. Such systems have been particularly important and successful in the domain of industrial process control where conventional software and teams of operators were unable to cope with the demands presented by rapidly changing, complex environments [13]. However as expert systems technology has proliferated and individual systems have increased in size and complexity, new problems and limitations have been noted [20], [24].:

- **Scaleability:** the complexity of an expert system may rise faster than the complexity of the domain.
- **Versatility:** a complex application may require the combination of multiple problem solving paradigms.
- **Reusability:** several applications may have requirements for similar expertise. In a conventional system this has to be coded afresh in each new situation.
- **Brittleness:** expert systems typically have a very narrow range of expertise and are generally very poor at identifying problems which fall outside their scope.
- **Inconsistency:** As knowledge bases increase in size, it becomes correspondingly more difficult to ensure that the knowledge they embody remains consistent and valid.

One approach designed to circumvent these shortcomings and satisfy the ever increasing demands for speed, reliability and integration is to compartmentalize the

problem solving into smaller, more manageable components and allow them to communicate and cooperate with each other (i.e. Distributed AI[2], [12][14]). In such systems knowledge, resources, control and authority are distributed amongst community members who then work together, in a coordinated and coherent manner, to solve one or several problems.

The ARCHON¹ (ARchitecture for Cooperative Heterogeneous ON-line systems) project [16], [22] is concerned with the development of a framework which enables multiple problem solvers (some of which may be pre-existing) to be interconnected so that they can communicate and cooperate with each other whilst solving complex, real-world problems. The real-world problems being tackled in the project are in the field of industrial systems: electricity management, cement factory control, robotics and control of a particle accelerator.

Within the field of Distributed AI many experimental platforms have already been built and these broadly fall into two categories:

- systems which test a specific type of problem solver, a specific coordination technique or a particular domain

eg. ETHER [18], ECO [9], DVMT [19], ATC [4]

- systems which aim to be "general" to some degree

eg. MACE [11], MICE [7], Cooper A [1]

However the weakness of these "general" systems is that they were not intended to be used in real-size problems (i.e. they lack the necessary power in a very narrow domain, but they are not generalizable. One of the major objectives of the ARCHON project is, therefore, to bridge this gap - to construct a cooperation framework which is both general enough and powerful enough to be used in a wide range of real-world industrial applications.

The remainder of this paper describes a typical ARCHON application in the field of industrial process control (electricity management) and highlights the characteristics of this domain which serve as important design forces. Section three gives an overview of the ARCHON architecture detailing the functionality of the various components and relating them to the problems identified in the previous section. Finally, section four describes the underlying philosophy of our approach which leads to a system which is both general and powerful.

2. Industrial process control

2.1 An Exemplar Problem

The two main applications within the project are concerned with electricity management and so a suitably simplified scenario (due to space limitations) will be used to illustrate the opportunities and benefits for cooperative interaction between problem solvers in this domain.

¹ The Archontes were the chief magistrates in Athens around 700BC. At that time there were nine, but with not very clear cut responsibilities and duties. These Archontes can be said to have formed a loosely coupled cooperative system with fuzzy boundaries for controlling the state.

The interaction takes place between three problems solvers, each of which is an expert system in its own right (see figure 1). The low voltage diagnosis expert system is capable of detecting and diagnosing faults based on information (network status messages and customer telephone calls) about the low voltage network which it receives. There is also an expert system which carries out the same role in the high voltage network. The third expert system is capable of receiving information about the weather from various sensors around the country and of predicting the weather in the near future or of recalling (estimating) the weather conditions in a certain area at a certain time.

These three systems were developed at different times, by different groups of people and employ different problem solving techniques. However they all share some common ground which means there is potential for them to interact. The link between high and low voltage systems is network connectivity: the low voltage network being fed exclusively from the higher voltage network. So if there is no voltage at the higher level then there will also be none at the lower level, meaning there is no point in the low voltage system searching for a fault. The weather monitoring system is able to provide useful information for the diagnosing process because one of the main causes of faults (on both networks) is damaged power lines and the major cause of such damage is bad weather (either lightning storms striking plant equipment or strong winds causing equipment to be blown over). If the weather monitor can provide information regarding bad weather then this may serve as a focus for the problem solving process or be used to increase the certainty of hypotheses of equipment near a major storm.

At present the information generated by the three systems is merely presented to the operator who has to collate it and draw appropriate inferences, a very demanding task during an emergency. This burden is increased substantially if the individual sub-systems produce diagnoses which are inconsistent. Therefore it is desirable that the three systems work together, sharing information and results directly so that the operator is freed from the drudgery of collating reports and resolving conflicts and can concentrate on more cognitive, strategic-level tasks.

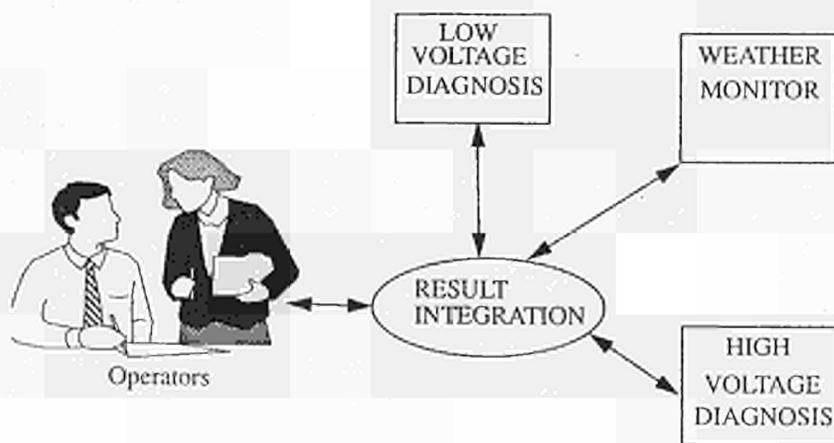


Fig. 1: Cooperative problem solving in electricity management.

The objective of ARCHON is to provide a framework into which these systems may be integrated, enabling the potential for cooperative problem solving to be realised. In the above example, the main benefits include decreased operator workload, better utilization of problem solving resource (eg. low voltage diagnosis system will not waste time trying to diagnose a fault which has been caused at the higher level) and increased confidence in hypotheses (since results have been cross-checked from several sources).

2.2. Major Design Forces

When targeting a system at a particular domain it is important to identify those characteristics which are likely to have greatest impact upon the design. After a careful study of industrial systems the following major design forces were identified:

- There are a large number of existing software systems, each capable of solving some aspect of the overall problem using a paradigm which is most appropriate for that activity.
- Typically one (or more) operator(s) has to combine the possibly inconsistent results of several sub-systems in order to make informed decisions about the process as a whole.
- Typical applications are composed of many different types of generic function which have been automated in many different ways [13]:
 - *Diagnosis*: delivering an understanding of a world state given some information about this world
 - *Planning*: sequence a set of possible actions
 - *Control*: particular case of planning where actions are executable and low-level
 - *Supervision*: reflecting decision link between diagnosis of a dynamic system and the alternative actions needed for exceptional situation handling

Having identified these design forces it is important to analyse their effects. Firstly, the majority of Distributed AI systems make little or no attempt to integrate pre-existing systems. However in a domain in which there has been substantial investment in automation (such as industrial systems) it is clearly desirable from an economic viewpoint to be able to interconnect such systems. Such re-use has the additional benefits that software productivity can be substantially improved [3] and dependability can be increased (due to greater and more diverse usage). It is unlikely that such integration will be possible without *any* modification to the sub-system, but we aim to minimize the changes necessary. Pre-existence also implies that sub-systems are likely to be fairly sophisticated in nature (cf nodes of a neural network and Blackboard Knowledge Sources [8], [15]) and will be capable of a significant problem solving in their own right (they represent a *cohesive* body of knowledge and problem solving capability).

The desire to incorporate pre-existing systems combined with the nature and complexity of the domain means that problems of heterogeneity can occur at many different levels [21]:

- **Hardware Platform**: eg. SUNs, Symbolics, VAXs
- **Operating System**: eg. UNIX, VMS, MS-DOS
- **Programming Language**: eg. LISP, Prolog, C, KEE, ART
- **System Type**: eg. classical control, databases, expert systems

- **Architecture:** Expert Systems - Frames, Blackboards
- **Semantics:** Understanding and distribution of concepts
- **Purpose:** Distribution of capabilities

Typically Distributed AI work has made simplifying assumptions about the uniformity of problem solvers or their domain of application, but as the above classification highlights heterogeneity is a pervasive phenomenon in real world problems. We argue that the form of heterogeneity which has the most significant impact upon the design of a cooperation framework concerns semantics and purpose [21], and propose that sophisticated agent and information modelling facilities (see section 3) are required to cope with these problems.

Many of the typical advantages of distributing both data and control when problem solving (as expressed in [2], [12] [14]) also apply to ARCHON applications; however the predominant benefits in this type of environment include:

- Enhanced problem solving
 - by sharing knowledge and data, systems may be able to make use of information which would not have been available normally (due to resource limitations, for example) and thus problem solving may be faster, of a higher quality, more accurate and so on.
- Ease the burden on the operator
 - in emergency situations operators are typically faced with a barrage of possibly contradictory information from multiple sources which they have to collate, interpret and act upon. By allowing the sub-systems to communicate with each other directly the operator's cognitive load is reduced considerably and he can concentrate on strategic level considerations.
- Increased reliability
 - because problem solving capability may be available at multiple sites within the community, failure at one node does not lead to total system collapse (i.e. system exhibits graceful degradation of performance).

3. Agent structure

The aim of the ARCHON framework is to create an environment in which social interaction is possible. However the problem solvers of figure 1 have no cooperative knowledge, they were conceived and built as stand alone systems and therefore do not know how to behave or participate in an environment which contains other entities. All they possess is the domain-level knowledge necessary to solve domain-level problems, e.g. how to diagnose faults in low and high voltage networks and how to predict the weather - we call such systems **Intelligent Systems**. In order for cooperative behaviour to be realised, the intelligent systems must be augmented with knowledge which enables them to engage in social activities (eg. to initiate, maintain and respond to cooperative situations, to be able to assess the needs of the community as well as

their role within the community and so on). Within ARCHON, this social awareness is achieved by enhancing each intelligent system with a series of modules embodying the necessary social knowledge². This collection of modules is collectively referred to as the **ARCHON layer** and the combination of an intelligent system and its ARCHON layer as an **agent**.

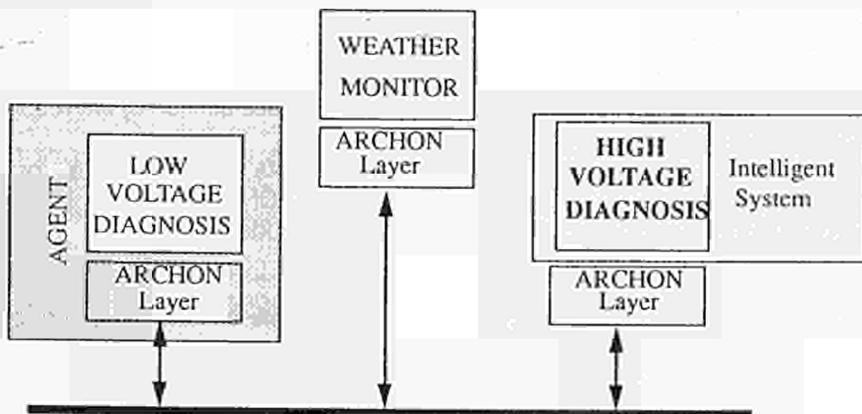


Fig. 2: An ARCHON community

The movement to a cooperative environment means a fundamental change in the requirements of an individual intelligent system, which the ARCHON layer must support. Whilst in the asocial situation depicted in figure 1, each system plans its activity exclusively on the basis of domain knowledge; in a social context an agent's activities are planned on the basis of both domain knowledge and social knowledge. An ARCHON agent, therefore, has two distinct (but related) roles to satisfy: that of a team member acting in a community of cooperating agents and that of an individual. Much of the early Distributed AI work concentrated almost exclusively on the former and paid scant regard to the latter; however ARCHON, and contemporary DAI in general [5], [6], places greater emphasis on the role of the individual. Therefore when designing the ARCHON layer both aspects should be catered for:

- Control (direct) local problem solving component
eg.. which tasks to launch, when they should be launched, their relative priorities and how best to interleave their execution, recovery from local exceptions
- Coordinate local activity with that of others within the community component.
eg.. when and how to initiate cooperative activity, which cooperation protocol (client-server, contract net, etc.) to employ, how to respond to cooperative initiations, which activities require synchronization.

² Using this approach means that the cooperative know-how is also conceived and implemented in a distributed manner. The alternative, one meta-system controlling all the others, was rejected because such a controller would become a computational and communication bottleneck and also because the complexity of the problem being tackled would prevent such a system from maintaining a consistent and complete view of the entire problem solving process (i.e. it would possess bounded rationality [23]).

These two perspectives provide a design rationale and separation of concerns upon which the ARCHON layer architecture can be based -see Figure 3. The monitor is responsible for controlling the local problem solving activity while the planning and coordination module (PCM) is responsible for controlling an agent's cooperative interactions. There is, however, a grey area between these two modules which has to deal with the impact of local decisions on the global perspective and of global decisions on the local activity - as choices about such interactions effect both the monitor and the PCM they are dealt with and agreed by both modules.

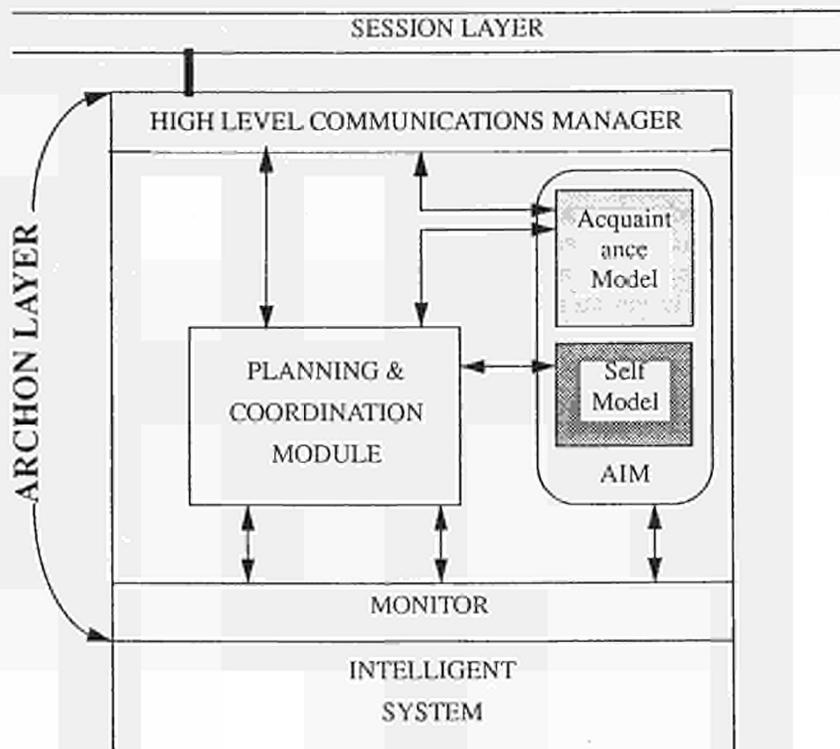


Fig. 3: Detailed structure of the ARCHON layer

The other components of the ARCHON layer are there to support these two modules. The High Level Communication Module (HLCM) supports the types of dialogue necessary for decentralized problem solving and coordination; offering facilities such as message scheduling, message filtering and intelligent addressing. The HLCM interfaces to an extended Session Layer³ which means ARCHON communities can be installed over networks conforming to the OSI standard. AIM (the Agent Information Management module) provides an object-orientated information model, a query and update language to define and manipulate the information and a distributed information access mechanism to support the remote access and sharing of information among agents.

3 The session layer has been extended to provide facilities appropriate for cooperative problem solving. For example, the standard does not cater for point to multi-point connections, something which is essential if broadcasts to groups of agents are required.

It is well acknowledged in many fields (including sociology, economics, politics and Distributed AI) that sophisticated cooperation requires participants to have some knowledge of other team members. The acquaintance models [2], [11], [21] provide exactly this knowledge - they are an abstract description of the other agents with which the modelling agent has to interact and are used by the PCM and the monitor for determining and mediating cooperative interactions. The type of plans, current state of processing, what information they can generate/are interested in and so on [21], [22]. The self model provides a meta-level description of the underlying intelligent system which the monitor can reason about when controlling the local system - the information to be maintained includes the status of active tasks, which tasks are pending, which tasks are waiting for which pieces of information, the relative priority of the various tasks, for example.

4. Achieving generality and power: a hybrid approach

By aiming to produce a cooperation framework which has a degree of generality, rather than constructing specialized systems for each particular application, it is important that the approach adopted is flexible yet still expressive enough. The two main weapons with which we attack this problem are both outlined in this section:

- Combining pre-compiled behaviours and reflective planning components to guide agent activities
- Using domain independent structures with domain dependent instantiations

One way of increasing the power of a system is to compile reasoning and action [17] - such "chunks" are usually referred to as behaviours. However systems in which all action is pre-compiled (reactive systems) have certain inherent limitations, the most pertinent of which is the fact that behaviours are usually specific to a certain problem or application. In contrast, systems which perform reflective reasoning and planning at run-time are applicable to a wider range of problems (particularly if this reasoning is based on domain independent structures - see below), but are typically more expensive in terms of the resources they consume. The ideal approach therefore, is to take the power offered by behaviours and combine it with reflective reasoning (generality) of the appropriate level. The characteristics of these two problem solving paradigms can then be used in the parts of the architecture to which they are best suited:

- Monitor: Requires fast response to large number of domain specific events and is difficult to generalise
 - Predominantly behaviour based
- PCM: Small number of social activities, need to reflect the particular run-time situation and offers greater scope for general descriptions.
 - Predominantly reflective

The second method is to make use of domain independent structures. Within the ARCHON architecture such structures include the communication facilities, the agent modelling facilities and the information modelling facilities (AIM). However due to space limitations the discussion will focus on the agent modelling facilities. The idea behind the acquaintance models is that they should embody most (ideally all) of the information necessary for supporting social activity and for evaluating the status of the community from the modelling agent's perspective. Similarly the self models should embody the

information necessary for controlling the underlying intelligent system and for assessing the agent's local situation. The semantics of these models are then given by a more or less generic control mechanism⁴. Hence generality is achieved because substantial parts of control mechanism can be common to all agents, but power is retained because domain complexity is represented in the models.

To illustrate this point we will return to the electricity management problem specified in section 2.1. The weather monitor's model of the low voltage diagnosis agent will represent the fact that it is interested in lightning strikes when trying to diagnose faults:

Modelling Agent: Weather Monitor
 Modelled Agent: Low Voltage Diagnosis Agent
 Slot Name: Interests
 Slot Structure: {<Name₁, Cond₁>,..... <Name_n, Cond_n>} (TEMPLATE)
 Slot Structure: {<LIGHTNING-STRIKES, (INSTANTIATION)
 WORKING-ON(DIAGNOSE-FAULT)>}

The semantics of this part of the agent model can be provided by the following generic control rule⁵:

IF knows(agent(?A), information(?I)) AND
 (∃ <Name_x, Cond_x> ∈ INTERESTS(agent(?A), acquaintance(?A2)):
 equal(Name_x, information(?I)) AND
 is-true(Cond_x))
 THEN can-help(agent(?A),
 acquaintance(?A2),
 send(agent(?A), acquaintance(?A2), information(?I)))

5. Conclusions

This paper serves as a mid-term progress report for the work currently being undertaken within the ARCHON project. It highlights those features of the project which distinguish it from previous work in the field of Distributed AI and hence defines the project's niche. The boundary of the range of problems being tackled has been described and an approach which leads to both a powerful and general framework has been outlined. The ARCHON architecture, as it exists at present, and the major design forces which give rise to this architecture have also been described.

At this stage in the project we are engaged on several simultaneous activities: specifying in greater detail the exact separation of concerns between the various components of the architecture, reconceptualising applications to ensure maximum utility is made of the cooperating systems metaphor and also verifying our architecture against the real world scenarios which exist within the project.

4 The underlying premise is that meaningful behaviour can be defined by the **structure** of the models and the actual contents merely provide the context for interpretation.

5 The meaning of the predicates is self explanatory. can-help(a1, a2, X) means that agent a1 is in a position to help acquaintance a2 by doing X. Whether it actually decides to do X will be based upon the current situations of a1, a2 and the rest of the community.

Acknowledgments

This paper represents the work of the whole ARCHON consortium which consists of the following partners: Krupp Atlas Elektronik, JRC Ispra, Framentec, QMW, IRIDIA, Iberduero, Labein, Electricity Research and Development Centre, Amber, Technical University of Athens, FWI University Amsterdam, Volmac, CERN and University of Porto.

In particular the following have contributed significantly to the current state of the ARCHON architecture: Abe Mamdani (QMW), Thies Wittig (Krupp) and Erick Gaussens (Framentec) to the overall architecture design; Claudia Roda (QMW) to the HLCM; Jochen Ehlers (Krupp) to the PCM; George Stassinopoulos, T Tsatsaros and M. Spyropoulou (all Univ. Athens) to the Session Layer Work; Francois Arlabosse, Daniel Gureghian and Jean Marc Loingtier (all Frametec) to work on the Monitor and Behaviours; Frank Tuijnman and Hamideh Afsarmanesh (both Univ. of Amsterdam) to work on AIM and Eugenio Oliveira and Long Qiegang (Univ. Porto) to work on generic rules.

References

- [1] AVOURIS, N.M., LIEDEKEKERKE, M.H.V. & SOMMARUGA, L. (1989). Evaluating the CooperA Experiment. Proc. of 9th Workshop on Distributed Artificial Intelligence, Seattle.
- [2] BOND, A.H. & GASSER, L. (1988). Readings in Distributed Artificial Intelligence. Morgan Kaufmann.
- [3] BROOKS, F.P. (1987). No Silver Bullet: Essence and Accidents of Software Engineering. Computer 20 (4) pp 10- 19.
- [4] CAMMARATA,S. McARTHUR, D. & STEEB, R. (1983). Strategies of Cooperation in Distributed Problem Solving. PRoc IJCAI 1983, pp 767- 770.
- [5] DEMAZEAU, Y & MULLER, J.P. (1990). Decentralized AI. Elsevier Science Publishers.
- [6] DEMAZEAU, Y & MULLER, J.P. (1990). Decentralized AI Vol II. Elsevier Science Publishers.
- [7] DURFEE, E.H. & MONTGOMERY, T.A. (1989). MICE: A Flexible Testbed for Intelligent Coordination Experiments. Proc of 9th Workshop on Distributed Artificial Intelligence, Seattle
- [8] ENGELMORE, R. & MORGAN, T. (1988). Blackboard Systems. Addison Wesley.
- [9] FERBER, J. (1989). Eco-Problem Solving: How to Solve a Problem by Interactions. Proc. of 9th Workshop on Distributed Artificial Intelligence, Seattle
- [10] FEIGENBAUM, E.A., McCORDUCK, P. & NII, H.P. (1988). The Rise of the Expert Company. Times Books.
- [11] GASSER, L., BRAGANZA, C. & HERMAN, N. (1988). Implementing Distributed AI Systems Using MACE. in [14].
- [12] GASSER, L. & HUHNS, M.N. (1990). Distributed Artificial Intelligence Vol. II. Pitman.
- [13] GAUSSENS, E. (1990). Needs and Opportunities for Expert Systems in the Process Control Field. Vacation School for Process Control, University of Strathclyde, Scotland.
- [14] HUHNS, M.N. (1989). Distributed Artificial Intelligence. Pitman.
- [15] JAGANNATHAN, V., DODHIAWALA, R. & BAUM, L.S. (1989). Blackboard Architectures and Applications. Academic Press.
- [16] JENNINGS, N.R. (1991). An Architecture for Cooperating Systems. Artificial Intelligence and Simulation of Behaviour Quarterly, Special Issue on Distributed Alm 76.
- [17] KISS, G. (1991). Layered Architectures for Intelligent Agents. IEE Colloquium on Intelligent Agents.

- [18] KORNFIELD, W.A. & HEWITT, C.E. (1981). The Scientific Community Metaphor. IEEE Trans. on SMC. 11, 1, pp 24- 33.
- [19] LESSER, V.R. & CORKILL, D. (1983). The Distributed Vehicle Monitoring Testbed: A Tool for Investigating Distributed Problem Solving Networks. AI Magazine, pp 15- 33.
- [20] PARTRIDGE, D. (1987). The Scope and limitations of First Generation Expert Systems. Future Generations Computer Systems, 3, 1, pp 1- 10.
- [21] RODA, C., JENNINGS, N.R. & MAMDANI, E.H. (1991). The Impact of Heterogeneity on Cooperating Agents. Proceedings AAAI Workshop on Cooperation among Heterogeneous Intelligent Systems, Anaheim, Los Angeles.
- [22] RODA, C., JENNINGS, N.R. & MAMDANI, E.H. (1990). ARCHON: A Cooperation Framework for Industrial Process Control in Cooperating Knowledge Based Systems 1990 (ed DEEN, S.M.), pp 95- 112, Springer Verlag.
- [23] SIMON, H.A. (1957). Models of Man. Wiley.
- [24] STEELS, L. (1985). Second Generation Expert Systems. Future Generations Computer Systems. 1, 4 pp 213- 221.

AN OPTICAL PHOTOREFRACTIVE CORRELATOR FOR ROBOTIC APPLICATIONS

H. RAJBENBACH, S. BANN, PH. REFREGIER, P. JOFFRE, J.P. HUIGNARD
Thomson-CSF, Laboratoire Central de Recherches, Domaine de Corbeville
F-91404 Orsay, France

H.ST. BUCHKREMER
KRUPP Forschungsinstitut, Fried Krupp GmbH, Munchener Strasse 100,
D-4300 Essen, Germany

A.S. JENSEN, E. RASMUSSEN
RISO National Laboratory, P.O. Box 49
DK-4000 Roskilde, Denmark

K.H. BRENNER, G. LOHMAN
Universitat Erlangen-Nurnberg
Physikalisches Institut Abt-V, Erwin-Rommel Str. 1
D-8520 Erlangen, Germany

SUMMARY

We have designed, implemented and tested a multichannel photorefractive optical joint transform correlator capable of performing sorting tasks for robotic applications. The use of mini-YAG lasers and liquid crystal spatial light modulators, in conjunction with updatable holographic BSO crystals has resulted in a compact correlator (600 x 300 x 300mm) with real time capabilities (100 msec recognition speed). Flexibility is a built-in feature and correlation is demonstrated for various applications. Electronic and optical preand post-processing for improving the demonstrator performances are also proposed.

1. INTRODUCTION

Optoelectronic processors that combine the high parallelism of optics with the accuracy of digital electronics have attracted considerable research interests for many years. In the last decade, a number of significant advances in the critical technologies involved in optoelectronic components have opened new opportunities for practical applications in image processing, pattern recognition, interconnects and high capacity storage. Today, high power compact solid state laser sources, high resolution liquid crystal spatial light modulators and dynamic holographic materials have come to sufficient maturity to be combined in powerful optoelectronic processors. The objective of the ESPRIT project "NAOPIA" (New architectures for optical processing in industrial applications) is to design and implement optical architectures for robotic applications and to demonstrate the potential of optics in solving massively parallel problems in the light of current and future material and device developments. The system to be described in this paper was designed for an industrial recognition task, namely to sort mechanical tools of different basic two-dimensional shapes. Optical correlators have several features that make them attractive in this class of applications. Early attempts

at pattern recognition with optical correlators relied on simple matched filters to distinguish objects(1). Low flexibility and severe alignment constraints suggest that more sophisticated schemes are needed. In this paper we report the realisation of a real-time multichannel joint transform correlator for a vision system(2). It is based on the use of real time holographic crystals for fast input update and relaxed alignment constraints. Besides the work described here, a prototype of a hybrid optical electronic vision system has been built. The prototype consists of a remapping device which maps the Fourier transform of an image into log-polar coordinates. The data in the log-polar mapped image are reduced to two 1D images which are recognized by an electronic correlator. This work is described in detail in the NAOPIA work report and in refS(3-4).

In the next section, we recall the optical architecture and theory of operation of joint transform correlators. In sect. III the optical system hardware is described. Sect. IV is devoted to its optoelectronic components and details their operating conditions. The application specification is given in Sect. V, followed by a presentation on the experimental system performances (Sect. VI). Finally, Sect. VII highlights the remaining problems to be solved with a brief discussion on the proposed solutions.

2. THEORY OF OPERATION

The optical processing scheme for multichannel joint correlation (5) between a reference set of objects $R(x,y)$ and an unknown object $S(x,y)$ is schematically described in Fig. 1, for a four-channel configuration. The input consists of two coherent images representing the reference $R(x-a,y)$ and the unknown object $S(x+a,y)$ shifted by a distance $2a$. The Fourier transform lens generates the following distribution $I(u,v)$ in its focal plane:

$$\begin{aligned}
 I(u,v) &= | \mathcal{F} [(R(x-a,y) + S(x+a,y))] |^2 \\
 &= | \tilde{R}(u,v) + \tilde{S}(u,v) |^2 \\
 &= \tilde{R}(u,v) \cdot \tilde{R}^*(u,v) + \tilde{S}(u,v) \cdot \tilde{S}^*(u,v) \\
 &\quad + \tilde{R}(u,v) \cdot \tilde{S}^*(u,v) + \tilde{S}(u,v) \cdot \tilde{R}^*(u,v)
 \end{aligned} \tag{1}$$

where \mathcal{F} is the Fourier transform operator and $\tilde{\cdot}$ denotes the Fourier transform of a function. The intensity distribution complex field products are then recorded in a real time non linear medium, a photorefractive crystal as will be described in the next section. The output of the joint transform correlator is given by performing an inverse Fourier transform on the recorded intensity distribution of eq(1).

Using the correlation theorem, by which the correlation of two images is equivalent to the Fourier transform of the multiplication of their complex field amplitude in the Fourier plane

$$R \otimes S = \mathcal{F}(\tilde{R}^* \cdot \tilde{S}), \tag{2}$$

the output of the multichannel correlator can be derived. By considering in addition that the two input images $R(x-a,y)$ and $S(x+a,y)$ are spatially shifted, we obtain the following output intensity distributions around the respective location coordinates :

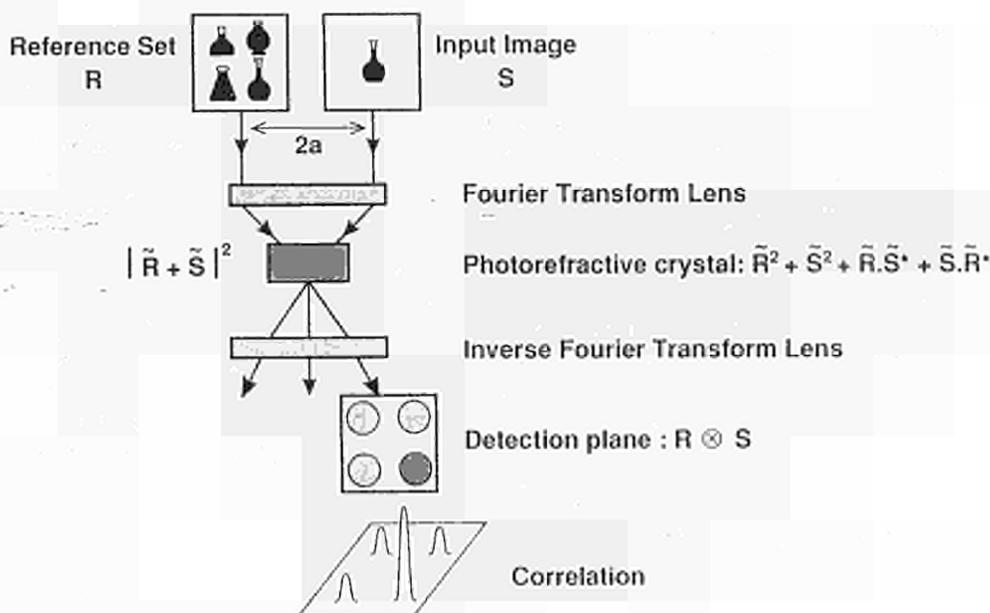


Fig.1 Optical architecture for a multichannel joint transform correlator. The input intensity distribution consisting of the reference set of objects R and the unknown object S is Fourier transformed. The interference between R and S is recorded in an updatable photorefractive crystal. Upon inverse Fourier transform, the output plane contains the correlation product $R(x,y) \otimes S(x,y)$.

The output is therefore divided in three regions. The side regions at $\pm 2a$ coordinates correspond to the last two terms in eq. 1 and contain the cross-correlation of the input images. The detection plane will be located in one of these two interesting regions. If the input object S is similar to one of the objects contained in the reference set, peaks of intensity will occur in the corresponding channel. The height of the peak in each channel (a window in the detection plane) is a measure of the degree of similarity of two objects (see fig. 1). In other words, the identification is performed by detecting the position and relative intensities of the correlation peaks in the corresponding windows of the output plane.

3. OPTICAL HARDWARE

Technology advances in optoelectronic components yields new possibilities for optics in many robotic applications that require speed and flexibility. The multichannel joint transform correlator described in this section benefits from many of these advances and was designed for industrial recognition tasks. The chosen application was the sorting of cutting tools manufactured by Krupp, with possible extension towards similar tasks identified by the other industrial partners of the project. Fig.2 schematically describes the optical hardware of the joint transform correlator. It operates as follows: the reference set of objects is presented to a first CCD video camera (1), while the unknown object S is viewed by a second camera (2). The two video signals are mixed so as to generate a single video image which represents the intensity distribution

$R(x-a,y) + S(x+a,y)$. This intensity distribution is relayed to a coherent optical beam with the use of a spatial light modulator (SLM) whose description and operation will be discussed in the next section. A coherent optical beam from a mini-YAG visible green laser is spatially expanded, reflects off the Mirrors M_1 and M_2 and is modulated by the image displayed on the SLM. A Fourier transform lens L_1 , located behind the SLM generates the complex Fourier transform field at the BSO material. The index modulation of this photo-refractive material is modified in real time proportional to the intensity distribution pattern. A real time phase volume hologram is therefore recorded (see next section). It is read out under Bragg direction by a plane wave from a red laser, e.g, a diode or a He-Ne laser. The diffracted field amplitude is extracted with the use of a dichotic mirror with high reflection coefficient in the red spectrum. Finally a CCD sensor in the Fourier plane of a second lens L_2 provides the correlation peaks displayed on a monitor or relayed to a personal computer.

The whole optical system, including the laser sources, the SLM, the photorefractive crystal was designed to be easily transportable and used in a pre-industrial environment. Its size is only 600 mm x 300 mm x 300 mm (L x W x D). This compact size design was allowed due to the use of modern optoelectronic components. It should be noted, however, that all of them are commercially available. Before discussing the performances of this joint transform correlator, a detailed description of the system components and their operating modes will be given in the next section.

4. OPTOELECTRONIC COMPONENTS

The size and performances of the multichannel joint transform correlator critically depend on its optoelectronic components characteristics. Although the system was designed as a research tool with emphasis on ease of access and adjustability, it was also considered that for further industrial development it is important to demonstrate its potential in terms of size, speed, robustness and flexibility. In this section we stress these parameters for the description of the components.

Non linear photorefractive BSO

Photorefractive crystals are attractive candidates for real time processing in coherent optical systems(6). In the joint transform correlator they are used to convert the Fourier transform field distribution of the SLM into a phase volume hologram. The formation of a phase volume hologram in these materials is due to the excitation and migration of photocarriers under local illumination. The corresponding photoinduced space charge field modulates the crystal refractive index through the linear electro-optic effect proportional to the light modulation pattern (fig.3). The index modulation can persist in the dark for several hours, but complete real time redistribution of the charges occurs when a new distribution illuminates the crystal, e.g., when new reference or signal is presented to the camera.

In the application presented here, the local illumination is due to the interference between the Fourier transform fields of the reference set $R(x-a,y)$ and the signal $S(x+a,y)$ displayed on the SLM. Therefore, the most important criteria to consider are:

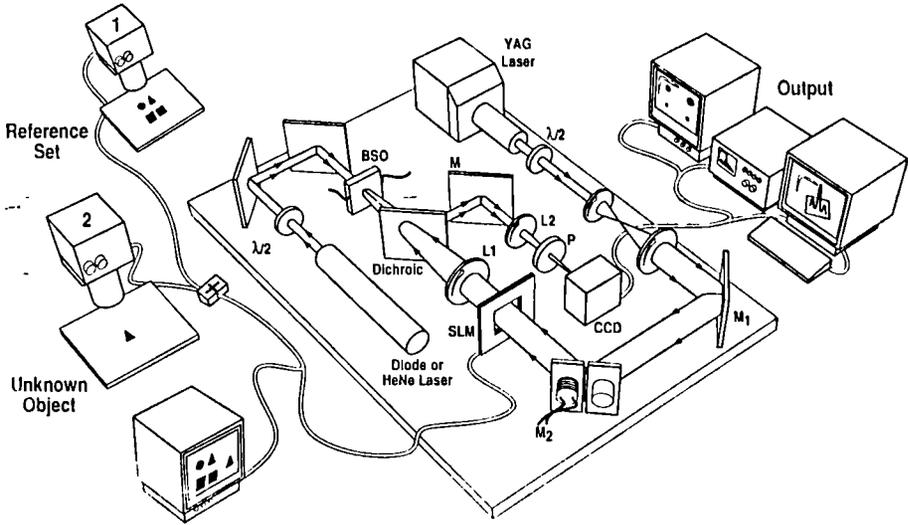


Fig.2: Optical layout of the multichannel photorefractive optical correlator. The reference set and unknown object are viewed by two CCD cameras, whose combined signals are relayed to a spatial light modulator (SLM). The coherent image carried on a visible YAG laser beam after transmission by the SLM is Fourier transformed by lens L_1 and recorded in the BSO crystal. An auxiliary diode or He-Ne laser reads out the photoinduced phase modulation in the BSO. The diffracted beam is extracted by a dichroic plate, and upon inverse Fourier transform by lens L_2 , provides the correlation product $R(x,y) \otimes S(x,y)$ and the CCD sensor.

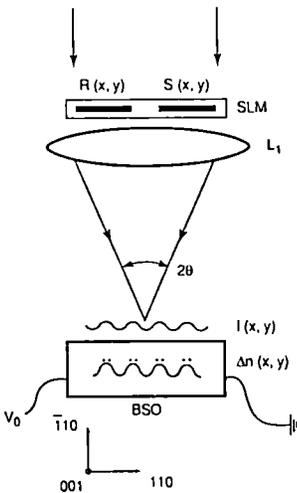


Fig.3: Mechanism for hologram recording in the photorefractive BSO by interference between the Fourier transform of the two images displayed on the SLM. Photocarriers are excited in the high illumination regions of the BSO and generate a space charge field which modulate index of refraction (Δn) through the electrooptic effect. The crystal is oriented for optimum signal-to-noise ratio after diffraction (electric field in the (110) direction)).

- (a) The response time τ : it should be of the order of the video rate $\tau_{\text{video}} \approx 30$ msec.
- (b) The diffraction efficiency: in photorefractive materials, the diffraction efficiency is known to be a sharp function of the angle between the interfering fields. This angle is however imposed by the distance between the two images ($2a$) and the focal length of the Fourier transform lens (see fig.3). For a 25 mm aperture SLM and a 300 mm focal length we obtain $2\theta = 5^\circ$. Only materials with high efficiency around this angle should be considered here.

5. APPLICATION SPECIFICATION

As mentioned in section III, the joint transform correlator was initially designed for industrial recognition tasks identified by KRUPP. The cutting tools to be sorted are formed in a variety of basic 2D geometric shapes. The tools are typically 15-20 mm across and 6 mm thick, with a hole drilled centrally to allow mounting onto a tool holder. During manufacture, the tools are required to undergo a coating process, for which they must be individually placed by a robot arm onto a circular platter in such a way that no two tools are touching. An efficient packing method can be used if the shape, position and orientation of each tool can be determined prior to placement on the platter. In this way, the process costs of tool manufacture can be cut, as the number of tools in each coating batch is increased. The basic specification of such a system is as follows

- 100% tool recognition
- maximum processing time ~ 100 msec
- x, y positional accuracy $\pm 500 \mu\text{m}$
- orientation accuracy $\pm 5^\circ$.

In addition, the system is required to be compact and flexible, i.e., new sets of reference tools or new objects should be possibly used without any modification of the optical hardware. The extension towards similar tasks should also be straightforward. Since the system has to be integrated in an industrial production line it has to be robust against the influence of disturbing variables emanating from the physical environment. The two main disturbances are mechanical vibrations and dust particles. The negative effect due to dust mainly can be avoided by an adequate enclosure of the system. Therefore, in the next section we focus on discussion of the system performances with respect to compactness, flexibility and stability.

6. DEMONSTRATOR PERFORMANCE

Optical hardware size

A picture of the correlator is shown in Fig.4. Individual components can be identified by referring to Fig.22. All the optical hardware holds on a 600 x 300 mm breadboard. In this first prototype demonstrator, most of the components are allowed to be individually aligned with the use of micropositioning holders. A further reduced size correlator can be designed after all the orientations and optimum positions of the components are known.

Discrimination characteristics

In a first experiment, we have tested the recognition performance of the correlator for the cutting tools. The reference set of objects is viewed by the first camera, and the four "unknown" objects are in turn presented (with optimum orientation) under the second camera. Fig.5 shows the experimental results. The left side is a picture of the image displayed on the SLM, while the right side is the output correlation plane as viewed by the CCD sensor and displayed on a monitor. The contrast of the monitor is adjusted for optimum peak visibility. No other post processing is used. The position of the correlation peaks unambiguously provides the information on the object shape and position. When the input object is moved, the correlation peaks move in the corresponding window thus confirming the real-time capabilities of the system. For the square and

Thin film transistor electrically addressed liquid crystal spatial light modulators satisfactorily fulfill these basic requirements. In these SLMs, each pixel is driven by an electrical transistor that controls the voltage across the liquid crystal cell, thus modulating by the electrooptic effect the amount of light flowing through. The optical efficiency is mainly governed by the size of the transistor relative to the pixel aperture. A good compromise between these parameters is found in the spatial light modulators used in video projectors. The SLM used in the joint transform correlator contains 320×264 pixels. Each pixel is $80 \times 80 \mu\text{m}^2$ wide, about 10% area of which is occupied by the transistor. The resulting SLM aperture is about 26×20 mm. This type of SLM is found in the Epson video projector model LC-500. Finally, the good phase flatness of these devices allows their use in the correlator.

Laser sources

To keep the system robust and compact it is important to operate with reliable solid state sources. Diode pumped YAG lasers offer an attractive alternative to ionized gas laser (Ar, Kr..). When frequency doubled with a non linear crystal such as KTP, they provide visible laser beam of high spectral and modal quality. A 80 mW, 532 nm wavelength, TEM₀₀ beam with more than a meter coherence length is available with mini-YAG lasers such as the ADLAS model 315c. The joint transform correlator utilises this laser. After modulation by the SLM and Fourier transform, about 10 mW is available for recording the real-time hologram. This intensity yields about 50 msec for the response time of the BSO. The readout red beam can be obtained with low power He-Ne lasers or semiconductor laser diodes. Although a successful operation of the correlator with visible semiconductor laser diode was demonstrated, the beam quality of these sources limits the sharpness of the correlation peaks. In the presented results we have used a 5 mW, 633 nm wavelength He-Ne laser.

Other components

The other components of the multichannel correlator are the following

- the Fourier transform lenses L_1 and L_2 from SORO are 300 and 200 mm focal length doublets respectively
- the CCD sensor in the detection plane (PULNIX). It contains $25 \cdot 10^4$ pixels. A transmissive red filter (not shown in Fig.2) is placed in front of the CCD and filters out parasite green light that reflects from most of the optical components.

Finally, to increase the correlation peak intensities, it may be important to increase the diffraction efficiency of the hologram written in the BSO crystal. A convenient mean capable of this improvement is known as the moving gating technique(10). It requires the interference of images $R(x,y)$ and $S(x,y)$ with a small frequency difference (typically $\delta f = 10\text{-}100$ Hz). In the joint transform correlator shown in Fig.2, the mirror M_2 is vertically split in two halves. The first half is fixed on a rigid holder, while the second half is mounted on a piezoelectric transducer. When the piezoelectric transducer is activated with a sawtooth voltage, the associated half of the green beam which reflects off this mirror experiences a Doppler frequency shift (see figure 2). By mean of this technique, the frequency of the beam carrying the reference set image $R(x,y)$ is shifted. This technique has resulted in a $\times 2$ improvement of the correlation peak intensities. In some cases and in particular when the unknown object has not its perfect same counterpart in the reference set, this technique can be used to improve the system performances.

(c) Wavelength sensitivity: the photorefractive crystal should be sensitive to the writing wavelength (green YAG laser) but the readout beam (from a red diode or He-Ne laser) should not erase the hologram.

Among all the photorefractive crystals available to date including LiNbO_3 , BaTiO_3 , KNbO_3 , GaAs , $\text{Bi}_{12}\text{SiO}_{20}$ (BSO), we have chosen to operate with BSO. In addition to a high sensitivity in the green-blue spectral range and low in the red(7), typical response time of about 50 msec are available for recording angles $2\Theta = 5^\circ$ with intensity levels of about 10 MW/cm^2 . In these conditions, diffraction efficiencies $\eta = 0.1 - 1\%$ can be obtained. An externally applied electric field on the crystal ($E_0 \approx 3\text{-}5 \text{ kV}$) is however needed to reach these specifications (drift mode recording). It is easily obtained with silver painted electrodes on the crystal.

Although photorefractive BaTiO_3 crystals are very attractive in terms of diffraction efficiency(6) (their electro-optic coefficients are very high) they were discarded here due to their rather slow response time ($\tau \approx 1 \text{ sec}$ with 10 mW/cm^2 incident intensity) and the large optimum image separation ($2a = 10 \text{ cm}$) that would required reference set and input image S to be displayed on two independent SLMS. The correlator stability would suffer from these constraints.

Finally, we should note that photorefractive semiconductors such as GaAs or InP are currently under study and could be used for joint transform correlators(8), but with near infrared laser wavelengths.

The material used in the presented multichannel joint transform correlator is a 1 mm thick BSO. An electric field $E_0 = 4 \text{ kv/cm}$ is applied along the (110) direction (inter-electrode distance 1 cm). In this electrooptic configuration leads a diffracted beam whose polarisation is rotated by 90° with respect to the polarisation of the incident beam. This allows high signal-to-noise ratios in the detection plane by correct orientation of a polarizer sheet(9) (P in Fig.2). Due to the large crystal thickness d compared to the average fringe spacing of the photoinduced hologram between R (x,y) and S(x,y), the readout beam direction has also to be adjusted at the correct Bragg incidence. The existence of a Bragg angular selectivity in thick crystals can restrict the usable field aperture in the reference set i.e., the number of channels of the joint transform correlator(9).

Liquid crystal spatial light modulator

The spatial light modulator is the key component to relay an image from a video camera to a coherent beam used for optical processing. The technologies involved in liquid crystal spatial light modulators have been developed for numerous consumer product oriented applications such as pocket televisions, portable computer screens, video projector... These technologies are becoming mature. High quality spatial light modulators are hence available for coherent optical processing systems such as the joint transform correlator. The basic requirements for these components are:

- (a) high resolution. The number of picture elements (pixels) in the image is a direct measure of the degree of parallelism used in the optical system. Today available SLM contains between 10^5 to 10^6 pixels arranged in square or rectangle formats.
- (b) small size. To limit the numerical aperture i.e. the size of the Fourier transform lenses, it is important to use moderate size SLM. The pixels should be as small as possible.
- (c) optical efficiency. The optical transmission coefficient of the SLM should be large to avoid the need of high power lasers in the correlator.

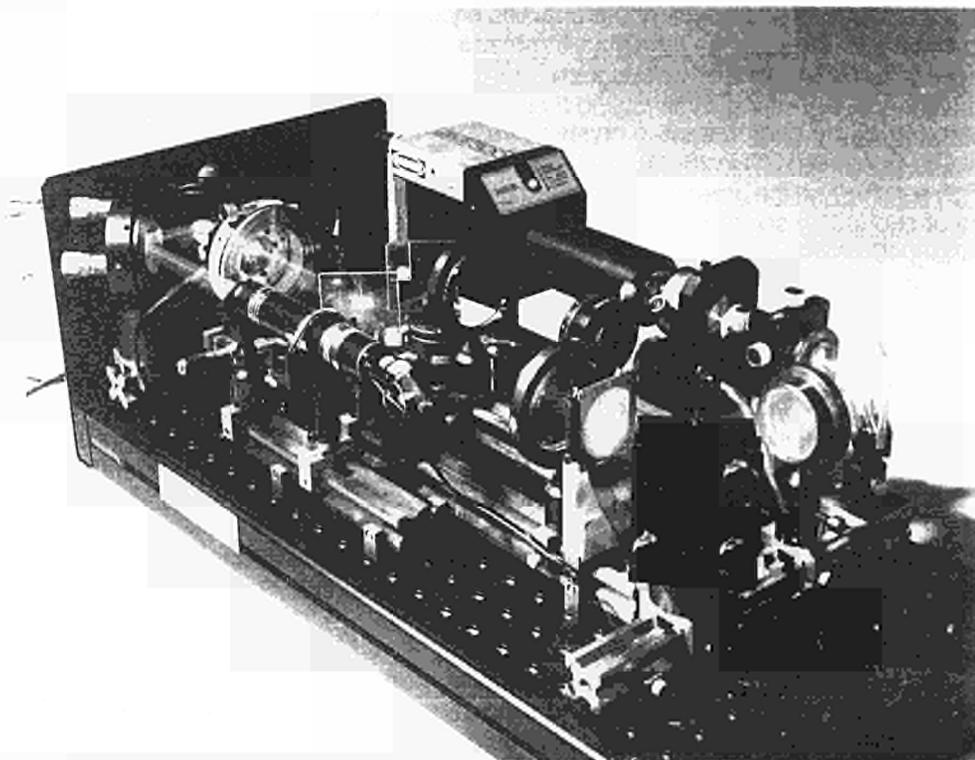


Fig. 4: Photograph of the multichannel joint transform correlator showing in particular: the 532 nm wavelength diode pumped YAG laser (in the back plane), the spatial light modulator (illuminated with the laser), and the photorefractive BSO in the focal plane of the Fourier transform lens. The red readout laser is in the front plane. The size of the demonstrator is 600 x 300 mm.

Reference	Input	Output

Fig. 5: Experimental results for tool sorting applications. The left sides show the reference set and unknown object displayed on the SLM. The right sides are corresponding video images from the CCD sensor in the correlation plane. The position of the brightest spot gives the shape and location of the unknown object. Note the presence of cross correlation peaks for two objects of very similar forms (square and rhombus). The bottom pictures show the correlator response to an object that does not have its counterpart in the reference set.

rhombus (look alike objects) note also the occurring of weak correlation peaks in two channels simultaneously. When an object not contained in the reference set is presented to the camera, correlation peaks may arise in few channels simultaneously, according to the similarity of its features with the reference objects. In the example shown by the lower picture in Fig.5 a screw bolt input generates weak correlation peaks in three of the four channels.

Angular selectivity and need of post processing

In practical applications, the unknown object is seldom aligned with its counterpart in the reference set. The dependance of the correlation peak intensity with respect to the angular shift between the two objects should be known for calibrating the processor. Anticipating the next section, an electronic post-processing can be used to determine the shape and angular parameters. Post-processing will consist of a look-up table or more sophisticated neural-like processing(ii). It is however important that the correlation peaks be above the noise level of the system. Fig.6 shows the evolution of the correlation peak intensities as a function of the angular shift between the objects. The angular tolerance ranges from 3 to 25°, depending on the object (3 dB criteria). Note also that the symmetry of the object give rise to periodic responses with 60°, 90°, 180° and 0° periods for the triangle, square, rhombus and circular objects respectively.

Flexibility

To demonstrate the flexibility of the multichannel joint transform correlator, we have performed sorting tasks for new sets of reference objects. A four-channel configuration is used for the bottle sorting application shown in Fig.7a whereas when using toy reduced models of different aircrafts the reference set is arranged in a three-channel configuration (Fig.7b). In all cases, the correlation peaks occur at the proper locations.

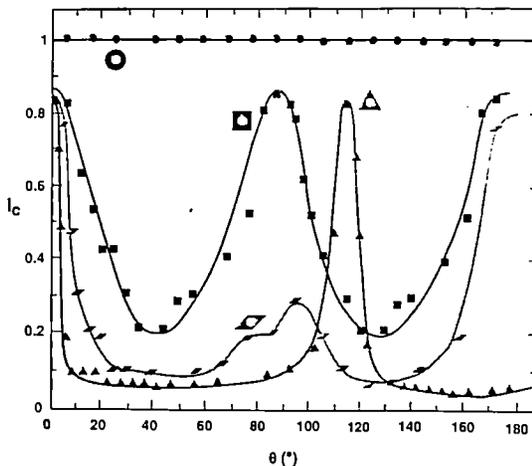


Fig.6: Correlation peak intensities as a function of the angular shift between the reference and input objects. The angular selectivity (3 dB criteria) depends on the objects, and ranges between 3 and 25°.

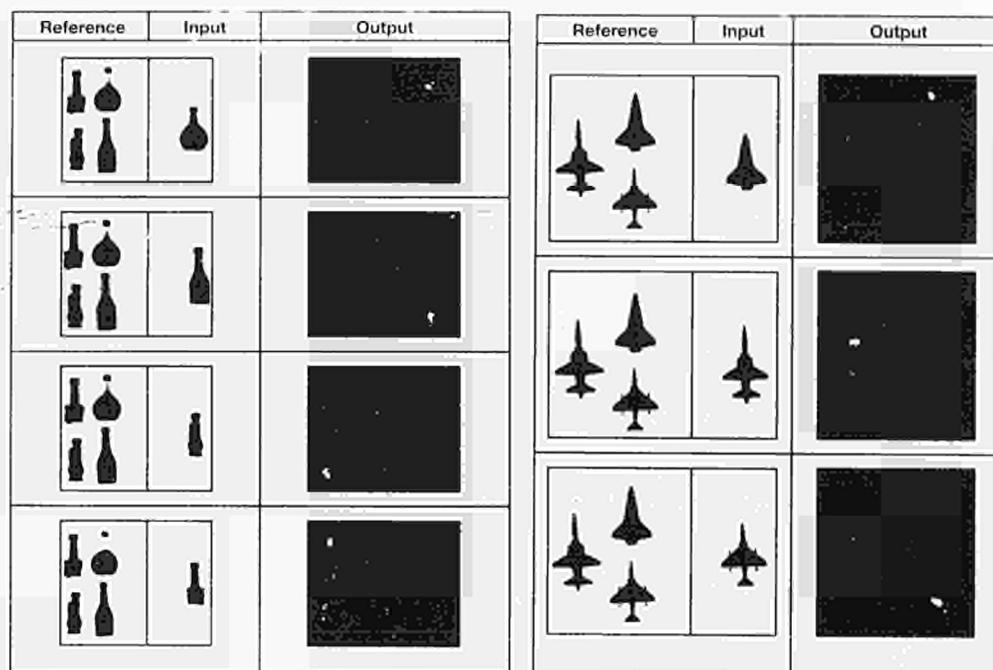


Fig.7: The joint transform correlator implements bottle (a) and aircraft toy model (b) recognition.

Response time and correlation peak intensity

Fig.8 shows the correlation peak intensity and the response time as a function of the electric field applied across the BSO crystal. The response time ranges between 35 and 100 msec corresponds to typical build-up times of photorefractive BSO at the intensity levels $\sim 10 \text{ mW/cm}^2$. Lower bounds can be readily used if low intensity correlation peaks are acceptable. The correlation peak intensity variation with applied voltage depicts a well known characteristic of photorefractive BSO(I0).

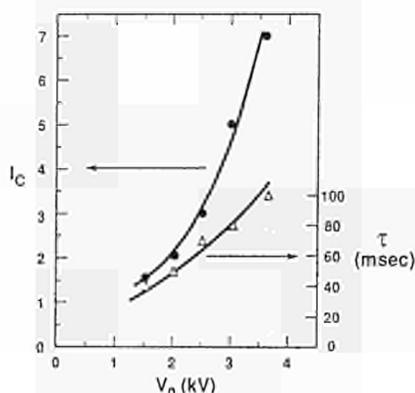


Fig. 8 : Correlation peak intensity and response time of the demonstrator as a function of the applied electric field on the BSO.

Noise resistance

In many applications, the unknown object $S(x,y)$ is corrupted by or buried in a strong background noise. It is centrally important that the correlation peak remains above a detectable level even for important noise levels. Although a quantitative analysis of the system performance degradation with added noise in the input image has not been carried out yet, we have observed the correlation peak intensity variation in the specific example shown in Fig.9. The top experimental result (no noise) yields a 60:1 measured peak to side lobe intensity ratio. The right picture represents the spatial intensity distribution across the horizontal direction in the correlation plane (it is obtained with a video line extractor). When noise is added to the input image, the correlation peak intensity decreases until it eventually disappears in the background noise. Note that in the bottom picture, the bottle in the input plane is already hardly visible. In the next section, morphological preprocessing is suggested to remove the noise in the input plane.

7. PERFORMANCE IMPROVEMENTS

The presented correlator is compact, flexible and operates at about video rate. It enables the recognition of rather simple objects when their angular position is close to their counterparts in the reference set. The remaining problems to be solved include rotation invariant recognition and noise robustness. Some of these issues were addressed in the course of the project and are briefly discussed in this section.

Noise removal by morphological preprocessing

Mathematical morphology is based on the notions of image dilation and erosion. It enables noise reduction, edge enhancement and object isolation. An optical implementation of morphological operation was proposed and simulation results were obtained by the University of Erlangen(12). Fig. 10 shows noise reduction obtained after iterative opening and closing morphological operations. By including morphology in a front-end processing unit, it is expected (although not demonstrated yet) that the joint transform correlator performance can be significantly improved.

Rotational invariant recognition

To enable successful recognition regardless of the object angular position (and still be able to determine its orientation) it is attractive to convert orientation parameters into shift parameters. An image mapping has to be performed. With a log-polar mapping of an image, the position, orientation and size will be expressed as shift parameters, or, to be more precise cyclic shifts(3-13). An optical log-polar transformer is designed and fabricated at RISO National Laboratory. This system opens the possibility for an optical vision system that is able to detect simultaneously not only the shapes and orientation of an object, but also its size. This last parameter determination, not considered in the joint transform correlator can greatly extend the performance and application range of the processor.

Electronic post-processing

In the joint transform correlator presented here, the shape of the unknown object is determined by the highest intensity correlation peak in the output plane, and its position

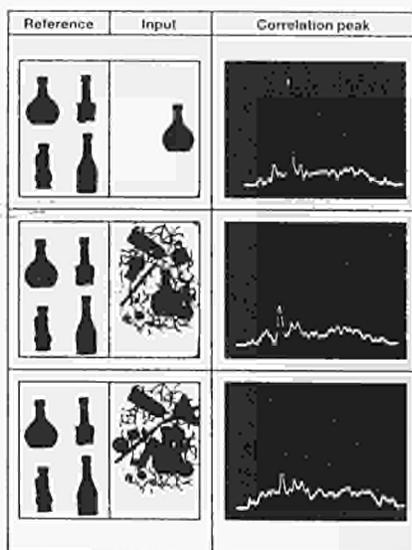


Fig.9 : When background noise is added to the input, the peak-to-side lobe ratio of the correlation decreases. The right side pictures are obtained by extracting a single video line in the correlation plane.

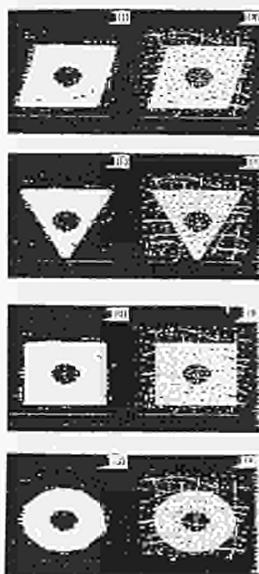


Fig.10 A proposed morphological front end processor removes the input noise using iterative closing and opening operations (simulation results).

by the peak location in the associated window.

It is attractive to incorporate a post-processing stage that will consider, not only the brightest correlation peak, but also the cross-correlations that occur in adjacent channels. This post-processing operates on a very limited amount of data. Three data per channel are relevant, i.e two for the positional information and one for the peak intensity. In such conditions it is important to consider electronic post-processing. A

powerful post-processing using neural approaches would enable to add a learning capabilities to the joint transform correlator (11). For example, we have previously shown (Fig.4) the correlator response to an input object not belonging to the reference set. The relative intensities in the four channels are characteristic of this object and can be used to train the neural network. In addition, post-processing neural approaches take naturally into account all the defects of the joint transform correlator. Work is progressing in that direction of Thomson-CSF.

8. CONCLUSION

A compact multichannel photorefractive optical joint transform correlator was designed, implemented and characterized. It operates with high performance optoelectronic components such as solid-state mini-YAG lasers, small size liquid crystal spatial light modulators and updatable holographic BSO crystals. It successfully performs many pattern recognition tasks, including the tool sorting for which it was initially designed. Its compactness, flexibility and speed were discussed. Some performance improvements were proposed and are currently under study.

(1) P.S.CAWTE, I.R.COOPER, G.C.GIBBON, S.C.WEBSTER, F.DUBOIS, M.BuCHEL, J.M.BRODIN, J.P.SCHNELL and B.LOISEAUX, "An optical image correlator for robotic applications", SPIE, vol. 1134, ECO2 Conference, 113438 (1989). (This paper reports the results of the ESPRIT II project 1035).

(2) H.RAJBENBACH, S.BANN, J.P.HUIGNARD, "A compact photorefractive joint transform correlator for industrial recognition tasks", Proceeding of the Optical Computing Conference, Optical Society of America, paper TUD5, Salt Lake City (1991)

(3) A.S.JENSEN and E.RASMUSSEN, "Invariant processing of optically mapped images with a one-dimensional correlator", submitted for publication in Opt.Eng., Oct. 1990

(4) A.S.JENSEN, E.RASMUSSEN, "New architectures for optical processing in Industrial Applications", RISO-M-2911, 1991, Technical Report on 1D optical vision system.

(5) C.S.WEAVER, J.W.GOODMAN, "Technique for optically convolving two functions", Appl. Opt. 5, 1248 (1966)

(6) P.GUNTER, J.P.HUIGNARD, (eds) "Photorefractive materials and their applications, 1, 11" Topics. Appl. Phys. vol.61 and 62 (Springer Berlin, Heidelberg, 1988)

(7) J.P.HUIGNARD, H.RAJBENBACH, PH.REFREGIER, L.SOLYMAR, "Wave mixing in photorefractive bismuth silicon oxide crystals and its applications", Opt. Eng.24, 586 (1985)

(8) G.GHEEN and LI-JEN CHENG, "Optical correlators with fast updating speed using photorefractive semiconductor materials", Appl. Opt.27, 2756 (1988)

(9) B.LOISEAUX, G.ILLIAQUER, JP.HUIGNARD, "Dynamic optical cross-correlator using a liquid crystal light valve and a Bismuth Silicon Oxide crystal in the Fourier Plane", Opt.Eng.24, 144 (1985)

(10) H.RAJBENBACH, J.P.HUIGNARD, B.LOISEAUX, "Spatial frequency dependence of the energy transfer in two-wave mixing experiments with BSO crystals", Opt.Comm.48, 247 (1983)

(11) J.FIGUE, PH.REFREGIER, H.RAJBENBACH, J.P.HUIGNARD, "Neural optoelectronic correlator for pattern recognition" to be presented at the SPIE annual meeting 21-6 July 1991, San Diego

(12) G.E.LOHMAN, K.H.BRENNER, "Optical morphological image processor", Proceeding of the optical computing conference, Optical Society of America TuB4, Salt Lake City (1991)

(13) A.S.JENSEN, L.LINDVOLD, E.RASMUSSEN, "Transformation of image position, rotation and size into shift parameters", Appl.Opt. 26, 1775 (1987)

A New Modular Course For Teaching About Software Engineering Measurement Within Academia

Martin Bush & Meg Russell
Centre for Systems and Software Engineering,
Department of Electrical & Electronic Engineering,
South Bank Polytechnic, Borough Road, London SE1 0AA, UK.

SUMMARY

The subject of software metrics is no longer new, yet there is a severe lack of educational materials, such as text books, available to teachers who want to include it within their courses. At least this is the opinion of the 140 academics who responded to an international survey undertaken by the ESPRIT project METKIT.

METKIT (Metrics Educational ToolKIT) is a three-year project aimed at raising awareness and increasing usage of software metrics (or, as we call it, software engineering measurement) within European industry by producing educational material aimed at both industrial and academic audiences.

This paper provides a description of the educational package which is being produced by METKIT for use within academia. It also briefly discusses the research that was carried out prior to the specification and implementation of the package, and the subsequent testing and dissemination of the package.

1. INTRODUCTION

The METKIT project began in February 1989 and is scheduled to last for three years. The project consortium comprises eight members, from four European countries, of which South Bank Polytechnic is the only academic institution. The other seven project members are:

- Brameur (UK) - the project leaders
- Sema (UK)
- British Telecom (UK)
- GMD (Germany)
- SES (Germany)
- Verilog (France)
- Dida*EI (Italy)

Each member of the consortium has expertise either in software engineering (including software engineering measurement) or in education generally, most having experience of both.

The main objectives of METKIT are as follows:

- to identify industrial needs in software engineering measurement and to determine the associated industrial and academic training requirements
- to rationalise existing knowledge in software engineering measurement
- to design and create educational material about software engineering measurement

- to evaluate the educational material in industrial and academic use.

The first year of METKIT was spent addressing the first two of these objectives. To address the first, two major international postal surveys were carried out, as well as a series of in-depth interviews. For the second, a conceptual framework was devised which now underpins the teaching material. These aspects are briefly described in the next section of this paper.

Having decided upon the use of a modular structure for the teaching material we then designed the structure of the material as a whole and wrote specifications for each of the modules in terms of teaching objectives and content. There are two broad "target audiences" which the METKIT material is being aimed at overall; managers and practising software engineers in industry, and students in academia. This paper deals specifically with the materials being produced for students in academia.

Now that most of the academic teaching material has been produced and tested (in house), and the package as a whole has been publicised, we are very encouraged by the response we have had and believe that it will have a significant impact towards getting the subject taught much more widely, and in a more balanced way.

2. RESULTS OF BACKGROUND STUDIES

Three major background studies were undertaken during the first year of the project. Two of these involved substantial international postal surveys and a series of in-depth interviews; one with industrialists and the other with academics. The third study consisted of a review of the subject area as a whole.

2.1 Postal Surveys and Interviews

One postal survey was designed to investigate the situation regarding the general awareness and use of software engineering measurement within industry. The results of this are outlined in [2]. The other survey was aimed at academia. Both surveys attempted to assess:

- the extent to which software engineering measurement is currently taught
- the use of various educational technologies and teaching methods in teaching software engineering in general
- the perceived needs for educational materials in software engineering measurement.

Our primary interest here is with the results of the academic survey. For this survey, approximately 450 identical 3-page questionnaires were sent to academic institutions across Europe, the USA and Japan. The survey was very successful in terms of the response rate achieved: 33% of recipients responded in total, with over 50% responding in both the UK and West Germany.

The survey revealed that software engineering measurement is already widely taught. Of those respondents whose departments taught software engineering, 69% taught software engineering measurement on at least one course. Details of the subjects covered and the courses concerned were collected. Many respondents reported that their teaching goals were hindered by a lack of available teaching materials for software engineering measurement; mostly they relied on general software engineering textbooks. When asked whether they felt that sufficient teaching aids exist, only 7% said "yes", while 60% said "no". A fuller description of the postal survey may be found in [9].

A series of interviews were also conducted to complement the survey. 10 interviews were carried out with academics and 11 with educators in industry. These interviews covered both current practices and future needs in some depth.

Interviews with teachers who were experienced in teaching software engineering measurement suggested that the best way to structure the subject matter was in a 'simple to complex' sequence. Motivation of students was considered to be all-important. It was widely recognised among those interviewed that students would not be motivated to use measurement unless they fully understood its purpose, and that it would be counter-productive to present individual measures or models out of context. It was seen as important to emphasise the importance of the proper use of statistics.

Most academic and industrial educators were using traditional teaching methods and materials. Teacher mediated group learning (ie. lectures or discussion) were the most popular. Use of modern educational technologies was more widespread in industry than in academia, but it was not felt that these new methods would take over from more traditional methods in the foreseeable future. Materials for self-study were particularly important to academics, whose students have study time outside lectures. The most popular materials for use here were books and software tools.

2.2 Review of the Subject Matter

The third major background study - the review of software metrics and related areas - led to the creation of a 'conceptual framework' that is intended to clarify and harmonise the various areas and approaches which make up software engineering measurement.

One of the aims of the background study was the identification of a boundary for the subject matter to be covered by METKIT. We soon realised that the term "software metrics", which had been the accepted term at the start of the project, is quite misleading in the sense that it does not cover all of the areas which we wanted to cover in METKIT. Strictly speaking, the term "software metrics" means *measures of software*. However, within METKIT we also deal with related areas such as the measurement and prediction of things like process cost, programmer productivity etc. We therefore adopted the term "software engineering measurement" instead, which we feel reflects this more accurately.

The conceptual framework is based on:

- a classification scheme that partitions the domain of software production by considering the entities that are amenable to measurement
- a rigorous approach to measurement extracted from basic ideas in measurement theory.

The framework is outlined below. A fuller description may be found in [1] or [3]. This framework is intended to act as an aid to understanding the subject, and it has strongly influenced the way in which the subject matter has been broken up into modules. It also underpins much of the teaching material itself.

Our view is that there are three fundamentally different kinds of entity - *processes*, *products* and *resources* which are involved in software development. A 'process' is an activity which results in one or more 'products'. In addition, a process requires 'resources' - such things as a hardware platform, software tools, personnel, office space etc. Hence an 'entity' can be an object, as in the case of products and resources, or an activity, as in the case of processes. With these basic building blocks, one can construct a life cycle model to represent any given software production system.

Figure 1 indicates the relationship between a process, the resources which it requires, and the products which are its results.

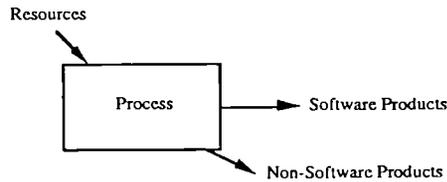


Fig. 1: Relationship Between Process, Products and Resources.

Some products are regarded as 'software' - we interpret this term widely, and include such things as source code, object code, specifications and user manuals. Others are 'non-software', by which we mean such things as time sheets, bug reports and minutes of meetings - anything that is produced during the process which is not software.

The software production system shown in figure 1 could represent the whole software production process or a part of it. In the latter case a life cycle model can be built up using a number of process, product and resource elements. Figure 2 gives a simple example. (Care must be taken in practice to define the semantics of any such model.)

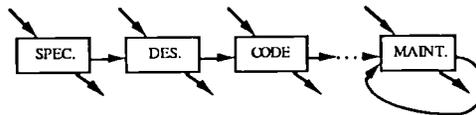


Fig. 2: A Simple Life-Cycle Model.

A minimum obligation for measurement is that we must know what entity is being measured, and we must understand the attribute (property) of the entity which we are attempting to characterise by measuring it.

In line with classical measurement theory (eg. [4], [8]) we define a *measure* as an empirical objective assignment of a number (or symbol) to an entity that characterises a particular attribute of the entity. We distinguish between *indirect measurement* of an attribute which is measurement that involves the measurement of other attributes of the same or other entities, and *direct measurement* which is measurement that does not depend on the measurement of any other attributes.

Indirect measurement involves modelling the attribute of interest in terms of other measurable attributes, in tree-like fashion. Quality models such as that of McCall [6] are examples of this kind of model, although they are rarely sufficiently detailed to enable quality to be measured objectively.

The models that underlie indirect measurement may be used for *assessment* or *prediction*. Whether such a model is being used for assessment or prediction depends on whether the attribute (and entity) to which the measure relates

- actually exists (in the case of products or resources) or has taken place (in the case of processes); or
- is yet to take place.

Quality models are normally of the former type; cost prediction models, on the other hand, are of the latter type.

3. THE MODULAR STRUCTURE

It was realised early on that the educational materials produced by METKIT would have to have some kind of modular structure. There were numerous reasons for this. These included the need to prepare overlapping materials for academia and industry without duplication of work, the need to produce courses that could easily be altered to suit individual organisations or include new developments in the subject and the need to provide materials to teachers who have widely differing amounts of time available.

A modular structure based upon simple-to-complex sequencing of the subject matter which took these factors into account was proposed for the educational material. This is described in detail in [5]. It is based on the "elaboration theory" of Charles Reigeluth [7]. Designs for all the academic METKIT materials have followed this structure.

METKIT is producing two teaching packages, corresponding to the two target audience groups (ie. managers and practising engineers, and academic students). Each package consists of a collection of modules. Each of these modules is a coherent whole of instruction, consisting of a number of sections dealing with different topics.

The academic package begins with an overview module which is divided into four sections. The fundamental ideas presented in this module are then expanded on in a number of 'lower level' modules. Each module expands the ideas in a particular section of the level above, although not all sections need be expanded. In order to study any module, the student must first have studied all modules which lie on the path up to the overview module - unless they have gained the knowledge and skills imparted by these modules from elsewhere. In addition to this there may be other modules which are prerequisite, but this type of dependency is minimised to allow flexibility.

Bearing in mind these rules, a teacher may construct a wide range of courses from the package, ranging in size from the overview module alone to the entire structure. Courses are effectively 'subtrees' of the total 'tree'.

The modular structure of the academic package is illustrated in figure 3, with some modules shown divided into sections. In the academic package there are four levels altogether. The introductory module (module AMO) contains approximately two hours of lecture material plus another hour of self study reading material. This in itself could be given as a short stand-alone introductory course.

Alternatively, a broad overview of the subject taking approximately 10 hours lecture time, plus up to a further 10 hours or so for self study, could be given using the introductory module plus the three modules on the level below. Given even more time, further modules could be added to make more detailed courses specialising in different areas. Of course the modules do not have to be taught all together. They could be spread throughout a broader-based software engineering course lasting several months or even years.

4. OVERVIEW OF THE MODULE CONTENTS

While studying the METKIT materials students are taught enough to enable them to answer the following high level questions:

- Why should we measure?
- What is measurement?

- What can we measure in software engineering?
- How can we measure it?

It is around these four questions that the METKIT packages as a whole are based, and the structure of the academic package clearly reflects this. As well as the successive levels of depth/detail, there is also a movement across the modular structure from theory to practice.

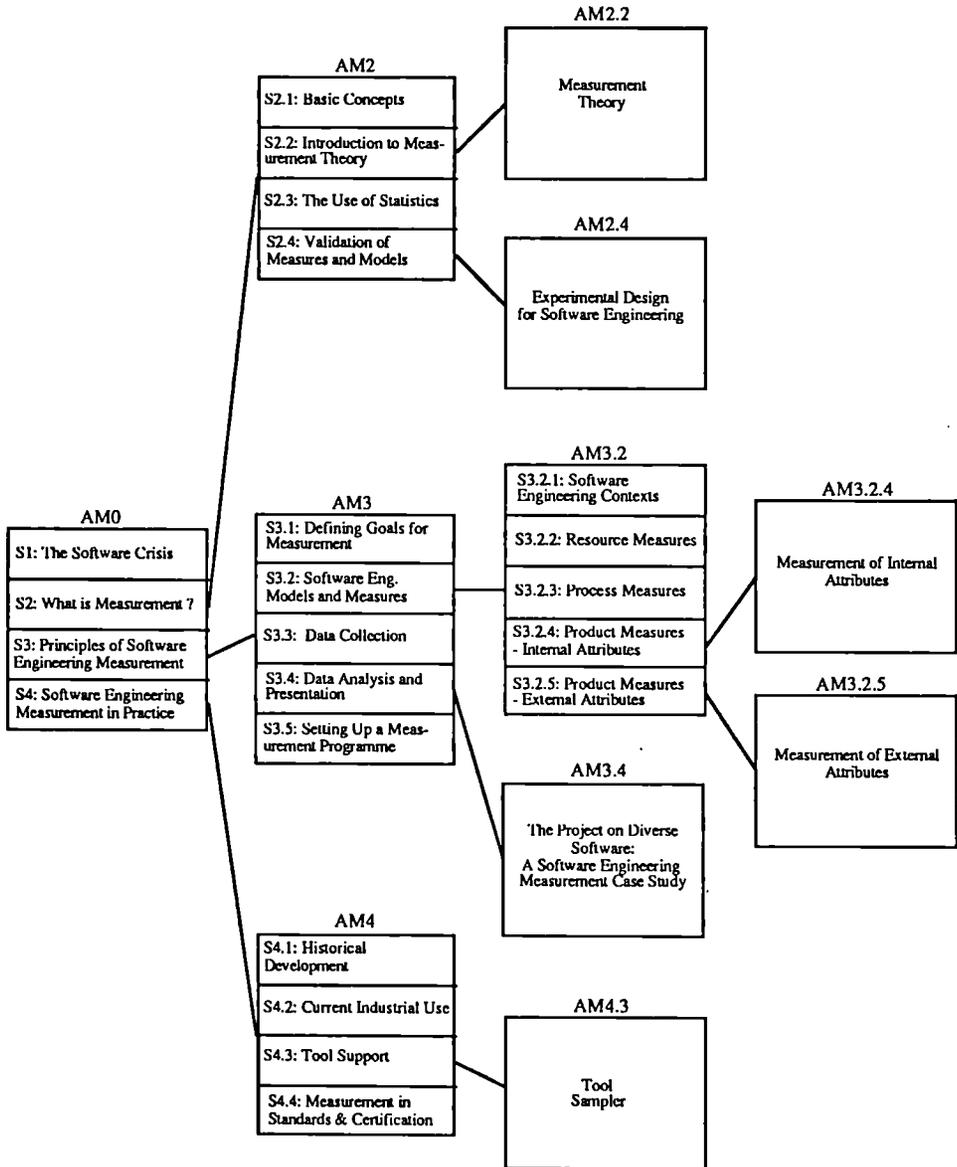


Fig. 3: Modular Structure for the Academic Package.

'Why should we measure?' is the question whose answers will offer the primary motivation to the students. This question is tackled explicitly in the first section of the overview module AM0 only, but is a recurrent theme throughout all of the modules, as motivation is felt to be the most important thing to be imparted to the students. The question is tackled in AM0 with a review of the software crisis and the ways in which measurement could help to solve it. Throughout all the modules it is emphasised that measurement is essential for assessment and prediction, or, as is more commonly stated, for "understanding and control".

'What is measurement?' is a question which has rarely been considered within software engineering, but which it is vital to consider before we can assess the state of the art in software engineering measurement, or begin to define measures and manipulate measurement results. This question is first considered in AM0, and this is expanded on in two lower level modules (AM2 and AM2.2), first without and then with mathematics. As well as being motivated to use measurement within software engineering it is important that the students learn to apply measurement in a meaningful way. The students will be taught to identify clearly what it is that is being measured, define their measure in an unambiguous way, and analyse the type of measurement data that they have in order to decide what statistical operations can meaningfully be applied. The students also learn to appreciate the need for properly designed experiments to validate prediction models and to make accurate comparisons between, for example, different methods of developing software. This is touched on in module AM2 and dealt with in more depth in module AM2.4.

'What can we measure in software engineering?' is also initially addressed in AM0, where the applicability of measurement to software engineering is demonstrated by way of a few simple examples. Throughout the package, but particularly in modules AM3 and AM3.2, it is stressed that measurement may usefully be applied to all entities in software engineering (ie. products, processes and resources). Each type of entity has a different range of attributes which may be measured. Two detailed modules, AM3.2.4 and AM3.2.5, deal comprehensively with various aspects of product measurement in particular.

'How can we measure it?' covers a wide range of issues. These include the need for a goal-driven approach to measurement, statistical issues including experimental design, data collection, data analysis and presentation, and the more managerial aspects of how to set up a software engineering measurement programme. All of these issues were felt to be very important by the experienced teachers who were interviewed, and provide crucial knowledge to anyone who is actually to carry out measurement and analyse the results. These issues are frequently skimmed over in existing courses. These issues are dealt with mostly in AM3. In addition, module AM3.4 contains a detailed real-life case study, including substantial exercises for the students in data analysis, presentation and interpretation.

In addition to supplying answers to these four important questions, the package includes materials on more general aspects of software engineering measurement in practice, which of course also help the student to answer the questions in a more informed way. This is covered in AM4. It was felt to be important to put existing research into context in terms of a formal measurement approach and the classification of software engineering entities into products, processes and resources. Although many existing measures (such as Halstead's and McCabe's) are often misused, students are likely to encounter them in the literature.

The material in AM4 is designed to reinforce the students' knowledge and enable them to consider past and current research in a critical light, whilst informing them of

the historical background to the subject. It is also valuable to the students to learn the extent of the use of measurement within software engineering in industry currently. The material also covers the availability and use of measurement tools (such as code analysers), use of software product and process standards, and certification.

5. PRESENTATION MEDIA

Following the investigations during the first year of the project, it was agreed that the METKIT educational materials would be largely designed for teacher mediated instruction. This method of instruction was the most popular amongst both academic and industrial educators. It is also considered to be the best method of presenting new concepts to students who have little understanding or experience of the subject (in this case software engineering measurement), and the best method for motivating students in a subject. The teacher mediated materials are, however, supported by materials for self study. This was recognised as being of particular importance to academic students.

5.1. Teacher Mediated Materials

Due to their popularity and flexibility, the academic modules are based around transparencies ('slides'). These are very widely used in both academia and industry, and hence most teachers already have the facilities to use them. Since many courses are already based on slides, the METKIT materials could very easily be integrated into these courses. Some teachers may want to put across a particular perspective, and it may be necessary to alter the materials to adapt to changes in the subject. A course based on slides offers the flexibility to do this as new slides may easily be produced and added, or existing ones altered. Also, some teachers may not want to teach all the material in a section in depth, as this may have been covered in another (non-METKIT) part of the course. With slides they can quickly skim over certain subjects, or even omit whole chunks.

In addition to the slides themselves we are also producing supporting materials for both the teacher and the student. Many of these materials will be used by both parties. They include various didactic guide-lines to make teaching and learning more effective as well as assessment materials, exercises, examples and references.

5.2. Self Study Materials

Various self study materials are being produced to support the taught modules. These are both text-based and computer-based.

The most important self study resource is a text book [3] which was partly commissioned by the METKIT project. This book is now available in bookshops throughout Europe. It was written in close collaboration with the project, and as a result its content closely matches the content of the modules, which in turn guide students by referring to the relevant pages of the book.

The introductory module is supported by a separate text. This is because the book is inappropriate as reading material for students at this stage, since it goes into rather more detail than is required. Furthermore, some students who will only study the introductory module may not wish to purchase the book.

Some of the materials from the industrial package will be integrated with the academic course at a later date. These may include video and computer-aided-learning modules. One of these is a rule-based system containing knowledge about software engineering

measurement for students to interrogate. Another is a "Tool Sampler" (which appears in figure 3 as module AM4.3), containing demonstration versions of various automated software engineering measurement tools so that students can acquire some hands-on experience with the kind of tools they can expect to find in industry.

6. ASSESSMENT OF THE ACADEMIC PACKAGE

The METKIT academic package is intended to be given by teachers who, although they must be familiar with software engineering, will probably not be experts in software engineering measurement. Consequently, testing is being carried out by teachers who are independent of the METKIT project, in order that the material can be properly assessed.

There are two rounds of assessment. The first, which took place earlier this year, involved testing some of the material on courses within South Bank Polytechnic. The results of these tests have already been analysed, and some substantial changes to the modules have now been implemented. The second round of tests, in the academic year 1991-92, will involve the revised modules, together with new ones, being taught in several academic institutions external to the project. The results of these tests are expected to lead to further changes before the finished product is available for release.

As well as being formally assessed for examination purposes, the students are also asked to fill in small tests and questionnaires relating to each module. The idea is to assess the materials for their effectiveness at achieving the educational objectives which we have set, and for their popularity with both students and teachers. This has required a significant amount of assessment material to be produced, and co-operation from both teachers and students in its completion.

7. DISSEMINATION

Being a technology transfer project, it is vital to the success of METKIT that as wide as possible an audience is made aware of the project, the educational materials and the subject of software engineering measurement in general. Throughout the early stages of the project we have tried to maintain as high a public profile as possible. This has involved presentations at conferences and workshops as well as circulation of publicity materials. The surveys in the first year also made hundreds of industrialists and academics aware of METKIT.

The project has two 'user panels', one academic and one industrial. These serve to review outputs of the project and provide fresh ideas. Members of these panels also provide test sites for the educational materials. Workshops have already been held for panel members, and these activities will continue, with 'train-the-trainer' sessions being a key feature for those involved in testing the materials. In particular, in August (1991) we are holding a two-day train-the-trainer workshop in the UK for European academics who will be testing the materials at other institutions.

Once testing of the materials is complete and changes have been made, the materials will be made widely available throughout Europe. We are currently investigating the contractual alternatives that will allow us to sell the materials to academics for a modest sum. The consortium also intends to continue with the development of additional teaching materials after the EC funding for the project has been exhausted.

ACKNOWLEDGEMENTS

We are very grateful to the Commission of the European Communities, who provide funding for METKIT under their ESPRIT II programme. The work described in this paper has been carried out in collaboration with colleagues both from within the South Bank Polytechnic and from our METKIT partner institutions. Thanks are also due to all those people who participated in the academic and industrial surveys.

REFERENCES

- [1] M. Bush & N. Fenton, *Software Measurement: A Conceptual Framework*, J. Syst. Software 12, p223-231, 1990.
- [2] E. Drummond-Tyler, *Industrial Needs and Practices*, METKIT report METKIT/SG/WP1.1/FR1, March 1990.
- [3] N. Fenton, *Software Metrics: A Rigorous Approach*, publ. Chapman & Hall, 1991.
- [4] L. Finkelstein, *Theory and Philosophy of Measurement*, in "Handbook of Measurement Science, Volume 1: Theoretical Fundamentals", P. Sydenham (ed.), Wiley, 1982.
- [5] L. Helwig, *Strategy for Building the Modular Structure of the METKIT Package*, METKIT report METKIT/DIDA/WP3.3/MOD.STRUC/LH, Jan 1990.
- [6] J. McCall, P. Richards, G. Walters, *Factors in Software Quality, Vols I, II, III*, US Rome Air Development Center Reports NTIS AD/A-049 014, 015, 055, 1977.
- [7] C. Reigeluth and F. Stein, *The Elaboration Theory of Instruction*, p335-381 of "Instructional Design Theories and Models: An Overview of their Current Status", C. Reigeluth (ed.), Lawrence Erlbaum Associates, 1983.
- [8] F. Roberts, *Measurement Theory with Applications to Decision Making, Utility and the Social Sciences*, publ. Addison Wesley, 1979.
- [9] M. Russell, *International Survey of Software Measurement Education and Training*, J. Syst. Software 12, p233-241, 1990.
- [10] R. Whitty, M. Bush & M. Russell, *METKIT and the ESPRIT Program*, J. Syst. Software 12, p219-221, 1990.

Databases and Executable Temporal Logic

Marcelo Finger, Peter McBrien, Richard Owens

Imperial College of Science, Technology and Medicine
Department of Computing
London SW7 2BZ
United Kingdom

Abstract

We present the means by which the executable temporal logic paradigm known as ‘the declarative past and imperative future’ can be used to describe the query and updating of databases. We discuss the implicit representation of time and evolving facts in a historical database, and how temporal logic is used as its query language. A set of temporal logic specifications is added to a historical database and the executable temporal logic paradigm is applied to the resulting system to model updates; this is achieved by a combination of the declarative and imperative readings of temporal formulae provided by this paradigm. The effects of altering of information about the past on the validity of past actions is described, and methods by which the database may be made consistent again with respect to the specification are outlined.

1 Introduction

Executable temporal logic is a relatively new product of the evolution of techniques for specifying and implementing computing systems. Traditionally, the language used for controlling the computer (i.e. the programming language) has been different from the language used to specify the required behaviour of the system. Programs consist of sequences of operations for the computer to perform, while specifications are descriptive, usually relating the output of the system with its input. The difficulty of verifying that programs behave correctly with respect to their specifications is essentially due to the clash between the *declarative* nature of the specification language, and the *imperative* nature of the programming language. The relationship between the specification and the program is made via a model of the program’s behaviour. The model has traditionally been some form of state transition system, with each state corresponding to a condition on some components of the program. Both the specification and the program tend to have structure which is of use during verification, but that structure is lost when relating the specification to the program through the model, which is essentially flat.

Since Kowalski demonstrated in 1974 the feasibility of giving imperative semantics to sentences written in a declarative language, the possibility exists of having a single formalism being both the specification and implementation language. The first such language was PROLOG, based on a subset of classical first-order predicate logic known as Horn clauses, and many alternatives have since been devised. One particular derivative of PROLOG of interest to relational database users is DATALOG [Ull88], which provides a set of restrictions on the standard definition of PROLOG so that the language may be easily interfaced to a RDMS (Relational Database Management System). A practical implementation of DATALOG has been made as part of the ESPRIT II project TEMPORA,

which provides an interface between BIM-PROLOG and Sybase RDMS so that operations on the PROLOG predicates are directly reflected by corresponding changes in the database.

A well known deficiency in the declarative specification PROLOG programs may provide is in their handling of updates and actions, such as inserting information into a database, or writing output to a terminal screen. Essentially the problem is caused by PROLOG only having one view or *model* of the world, with this model representing all the different situations or *states* that a program and its data may pass through during execution. Executable temporal logic addresses this problem by

- providing a series of models to represent the changes in data over time, and
- describing the update operations as conditions on past models which imply changes to be made to future models.

The first item is essentially the representation provided by *historical databases*, where facts are stored as *holding* at certain times and not others. Thus the model of executable temporal logic corresponds to that of historical databases, and the use of temporal logic to describe querying and update rules on the database would seem natural.

The descriptions of conditions on the past causing changes to the future are expressed in *temporal specifications*, which we may more simply term *rules*, of the form:

$$\text{past formula} \wedge \text{present formula} \Rightarrow \text{future formula}$$

A set of such rules constitutes a temporal logic program, and are executed by ensuring that at any time when the condition of a particular rule about the past and present holds, then the corresponding action of that rule is made to hold. Assuming a discrete model of time, we may simplistically specify the execution as:

1. We begin with a database Δ_0 , and a set of rules $\Gamma = \{query_i \rightarrow action_i\}$
2. At time t , find the set of rules $R \subseteq \Gamma$ for which $query_i$ holds against database Δ_t .
3. For each member $query_i \rightarrow action_i \in R$, make $action_i$ true, creating Δ_{t+1}
4. Repeat from step 2 for time $t + 1$

This paper sets out to explore the links between historical databases and executable temporal logic. In Section 2 we describe how data may be stored and queried in a temporal database, and give some simple examples of how update rules can be expressed in temporal logic. Section 3 describes the querying and update operations in a more formal manner. Finally, Section 4 gives some examples of action rules and deals with the procedures that must be followed when we make changes to information we held about the past, and thus need to review any actions we made based on that information to ensure that the declarative reading of our temporal logic rules is maintained.

2 Time in Databases

A common approach to the handling of time in databases is to consider temporal information to be just another attribute of relations, and state it explicitly as part of a relation when required. The relational database model has close similarity with first-order predicate calculus, DATALOG being a language which connects the two paradigms. When we deal with time added as attributes to relations in DATALOG programs the temporal information appears as additional arguments to predicates.

As an illustration, consider a database with the predicates $accounts(name,number)$ and $balance(number,value)$. We might choose to store the period $[s,e]$ each account is open as two additional arguments to the account predicate, but only store a record of the current balance of the account, and thus add no additional arguments. The database would then look something like:

```

account(peter, #1, 1/1/91, infinity)
account(james, #2, 3/7/89, infinity)
balance(#1, 200)
balance(#2, -1000)
    
```

Note that we store infinity, which represents here the maximum time permitted in a bounded model, as the end of the account since at this stage we do not know when the account will close.

When we come to make updates to the database, the use of predicate calculus to describe its behaviour means we can only achieve changes by means of side-effects, and therefore the updates have no logical semantics. For example, if in the above database we decide to close account #2 on 3/1/91, we will have to delete the information about balance(#2, -1000) and alter the information about account #2 to:

```

account(james, #2, 3/7/89, 3/1/91)
    
```

Here we cannot use `DATALOG` to describe in a declarative manner the deletion of the balance, since in predicate calculus there is no means by which we can move from one model to another, hence it is inconsistent to state that something is both true (account #2 has a balance) and not true (account #2 has no balance). Temporal logic overcomes this problem by providing (conceptually) multiple databases, each associated with a particular period of time. This has a direct equivalent in the historical database model where periods of validity are associated with each tuple or field of a relation. There are many alternative database models proposed in the literature for associating time values with the tuples or fields on a relation, but [TC90] has shown that any historical database with a linear discrete bounded model of time could be represented by a general *temporal structure*, and the queries can be described by an extension of the relational algebra with temporal operators. In [GM91] this approach was tailored to have a closer fit with the primitive operators on temporal logic, and the manner by which existing relational databases could be used for the storage and querying of temporal information was described.

Figure 1 illustrates an example database in the temporal structure, which shows the account balances and details changing with time. We need no longer make explicit reference to the time that an atomic sentence refers to within the sentence itself, but instead say that if the sentence holds at a particular time (i.e. it is in the database for that time), then it is true at that time.

31/12/90	1/1/91	2/1/91	3/1/91	4/1/91
account(james, #2) balance(#2, 400)	account(peter, #1) account(james, #2) balance(#1, 100) balance(#2, 300)	account(peter, #1) account(james, #2) balance(#1, 200) balance(#2, -1000)	account(peter, #1) account(james, #2) balance(#1, 200) balance(#2, -1000)	account(peter, #1) balance(#1, 200)

Figure 1: Temporal Structure of Banking Database

A practical point to note is that a more compact representation of the temporal structure is obtained by storing the *intervals* over which tuples hold. Each tuple of relation P holding over a series of consecutive states (or models) \mathcal{M}_s to \mathcal{M}_e inclusive can be represented as the tuple plus the interval $[s, e]$. When we want to see if the tuple holds at a given time t , we simply check that time t is within the interval associated with the tuple.

The example temporal structure of Figure 1 may be encoded using intervals to be:

Data Representation	validity
account(peter, #1)	1/1/91–infinity
account(james, #2)	7/7/89–3/1/91
balance(#1,200)	1/1/91–1/1/91
balance(#1,200)	1/1/91–1/1/91
balance(#2,400)	31/12/90–31/12/90
balance(#2,300)	1/1/91–1/1/91
balance(#2,-1000)	2/1/91–3/1/91
balance(#2,200)	4/1/91–infinity

A database in the temporal structure is called a *historical database*, which can sometimes be a misleading title since the database may contain information about the future as well as the past and present. To query such a database, we use a first-order version of the temporal logic USF [Gab89].

2.1 Introduction to the Temporal Logic USF

Temporal logics had their beginnings in *tense logics*, formal languages for the study of time in linguistics. Tense logics include two basic modalities, F and P , for accessibility in the future and in the past respectively. Intuitively, for some sentence a , Fa is read as ‘it will be the case that a ,’ and Pa is read as ‘it was the case that a ’.

The various topologies of time, and the modalities which range over those topologies, are what distinguish one temporal logic from another. Despite the philosophical debate about which particular topological characteristics should be used, computer scientists have been content to consider just a handful of such structures:

- linear discrete time temporal logics based on the integers \mathbf{Z}
- linear dense time temporal logics based on the rationals \mathbf{Q}
- linear continuous time temporal logics based on the reals \mathbf{R}
- branching discrete time temporal logics based on sequences of integers \mathbf{Z}
- branching dense/continuous time temporal logics based on the sequences of rationals \mathbf{Q} or reals \mathbf{R}

The choice of structure for a temporal logic dictates to a large extent the temporal connectives present in that logic. If we take an abstract view of a structure (also known as a *frame*) as a pair $\mathcal{F} = (S, \prec)$ where S is a set of states and \prec is a binary relationship over those states, we can motivate the use of particular connectives. Most temporal logics tend to have irreflexivity and transitivity as constraints on \prec , and we shall follow this. Each formula involving a connective is evaluated with respect to some state which is an element of S . In the following, the formulae are evaluated with respect to $s \in S$.

The \Diamond connective, when applied to a formula φ , shifts evaluation of φ from s to some state t where $s \prec t$. Intuitively $\Diamond\varphi$ is true in state s if φ is true in state t , $s \prec t$. If we make the frame concrete by selecting a particular set to be S , and a particular relationship to be \prec , we can give meaning to \Diamond . Let $\mathcal{F} = (\mathbf{N}, \prec)$, so that our set of states is the natural numbers, and the binary relationship is the ‘less than’ ordering. So $\Diamond\varphi$ is true in state s if φ is true in state t , where $s < t$. If we are using the natural numbers to model time, then a state t where $s < t$ is in the future of state s , hence $\Diamond\varphi$ is true in a state if φ is true in some future state. Thus the \Diamond connective is essentially the F connective of tense logic. There is a dual connective for \Diamond , namely \Box , which is defined as $\neg\Diamond\neg$. Thus $\Box\varphi$ is true in a state if $\neg\Diamond\neg\varphi$ is true in that state, so that it is not the case that $\neg\varphi$ is true in some future state, i.e. φ is true in all future states.

So much for states in the future. States in the past are accessed by the \blacklozenge connective, where $\blacklozenge\varphi$ is true in state s if φ is true in state t , where $t < s$, i.e. state t is in the past of s . Hence $\blacklozenge\varphi$ is true in a state if φ is true in some past state. The dual of \blacklozenge , namely \blacksquare , is defined as $\neg\blacklozenge\neg$. Thus $\blacksquare\varphi$ is true in a state if it is not the case that $\neg\varphi$ is true in some past state, i.e. φ is true in all past states.

It can be shown that the temporal logic that we have illustrated so far is not as expressive as a first-order logic which describes time explicitly. In fact we must introduce a new binary modality \mathcal{U} (until), together with its past-time equivalent \mathcal{S} (since), from which all other connectives can be derived. For $a\mathcal{U}b$ to be true means that there immediately follows a sequence of states in which a is true; this sequence must be terminated by a state in which b is true. Thus a is true *until* b is true. Similarly for the past-time equivalent \mathcal{S} , we have $a\mathcal{S}b$ true when we are at the end of a sequence of states in which a was true, and the sequence was started by a state in which b was true.

The logic USF contains, therefore, the following connectives:

$\blacklozenge A$	A holds in some previous state
$\blacksquare A$	A holds in all previous states
$\bullet A$	A holds in the preceding state
$\blacklozenge A$	A holds in some future state
$\square A$	A holds in all future states
$\circ A$	A holds in the next state
ASB	A holds in all the states since B holds
$A\mathcal{U}B$	A holds in all the states until B holds

2.2 Temporal Queries

In USF, the flow of time is viewed as a sequence of first-order models, every model being a state of the world at a certain time. The relations of the relational database (RDB) are mapped to the predicates in USF in exactly the same manner as is done for DATALOG, except queries have associated with them the time at which they are to be evaluated. A tuple of a relation P with interval $[s,e]$ in the database holds as an atomic sentence with predicate P at time t (in model \mathcal{M}_t) if $s \leq t \leq e$. Thus we can view the database as having the temporal structure, whilst storing the information more compactly using intervals.

It has been shown [GM91] that we can define operators similar to \mathcal{S} and \mathcal{U} in an extended relational algebra termed the *temporal relational algebra*. As an example of the use of these operators, consider the following query to the database in Figure 1, made with reference to time 4/1/91, which finds the name of all accounts open since 1/6/90. (The answer assumes that james has an account in each box off to the left of the diagram going back to at least 2/6/90. The underscore indicates a variable whose value is irrelevant to the result of the query.)

Query: $\text{account}(X, _)\mathcal{S}\text{time}(1/6/90)$

Answer: $X = \text{james}$.

Since the primitive operators are fully first-order expressive we may use them to query a historical database knowing the expressive power provided is akin to that provided by the relational algebra (and hence SQL) on the relational model. We give here a few example queries on the database in Figure 1. Suppose we are at 4/1/91 and we want to know which accounts were closed today. We can do so by finding the accounts which existed during the previous day (using the \bullet operator) and checking that they no longer exist today.

Query: $\bullet \text{account}(X, Y) \wedge \neg \text{account}(X, Y)$

Answer: $X = \text{james}, Y = \#2$

If we want to know who are the clients that in the past had a negative balance, we can use \blacklozenge to search for all the previous balances with a value below zero:

Query: $\blacklozenge(\text{balance}(Y,B) \wedge B < 0 \wedge \text{account}(X,Y))$
 Answer: $Y = \#2, B = -1000, X = \text{james}$

From this short introduction, it can be seen that we can use temporal logic to both model and query the information in a historical database. However, we may also use executable rules to describe updates.

2.3 Steps Toward Executable Temporal Logic

In general the models for temporal logic formulae are infinite, and so the earliest theorem-provers relied on the *finite model property* possessed by many of the logics of interest. Essentially this property states that if a formula is satisfiable then there is a model for the formula which is periodic in nature, and can thus be described by a finite model. Procedures based on this are simple to implement, although unfortunately they are of high complexity. It is clear that without significant re-engineering this approach is not suitable for implementing a programming language such as a temporal version of PROLOG. As is the case with classical logic, a resolution-based method of checking the satisfiability of a formula was required.

Abadi and Manna [AM90] devised just such a resolution method for a non-clausal temporal logic which extended the non-clausal resolution method for ordinary first-order logic to handle quantifiers and temporal operators. This led to the development of TEMPLOG, which is perhaps the best developed example of a PROLOG-like programming language based on temporal logic [AM87]. It is characterised by its use of time structures based on the natural numbers, and its use of flexible predicates.

Gabbay's TEMPORAL PROLOG [Gab87] extends the PROLOG definition of a clause to "ordinary clauses" which are true for a single state, and "always" clauses, which are true in all states. Each ordinary clause has a head which is a conjunction of atomic formulae and formulae of the form $\blacklozenge A$ or $\blacklozenge A$ where A is a conjunction of ordinary clauses. The body of an ordinary clause is either atomic, a conjunction of bodies, or a formula of the form $\blacklozenge A$ or $\blacklozenge A$ where A is a body.

TEMPURA [Mos86], [Hal87] is considered to be one of the furthest developed temporal programming languages. It is based on interval temporal logic. Variables in TEMPURA are considered to be flexible, and the language provides a host of facilities to assign values to the variables over time. In many respects, TEMPURA is close to traditional programming languages such as Pascal, but has very clean logical semantics.

We come now to the recent attempts to interpret propositional and full first-order temporal logic as a programming language. In [Gab89], Gabbay defined imperative semantics for the temporal logic USF which can be used to "animate" a temporal logic specification. Essentially executing USF formulae requires them to be transformed into a normal form, which is a conjunction of separated rules of the form

$$\text{past formula} \wedge \text{present formula} \Rightarrow \text{future formula}$$

Any set of USF formulae can be transformed into this form, as proved in [Gab89]. The execution of a temporal specification consists in checking the antecedent against the current state of a historical database and, in the case where the check succeeds, forcing the consequent to be true in the database. The notion of the imperative future regards the consequent not as a declarative condition to be checked, but as an imperative which must be *made* true.

A language which is essentially a restriction of USF has been defined and implemented as part of the TEMPORA project. The language allows queries using all of the temporal connectives presented

in Section 2.2, together with recursion in a PROLOG-like language. Actions are restricted to those with determinate semantics (i.e. $\bigcirc A$, $\square A$ and a restricted \mathcal{U} used in the form $A \mathcal{U} \text{time}(e)$ to name a specific interval over which A is made to hold). The data model is restricted to a finite bounded one, to allow for the easy mapping of queries onto a relational database model, and implementation is achieved by executing the rules in BIM-PROLOG with a tight coupling to a Sybase RDMS which reflects updates made to PROLOG predicates as database relation updates. The tight coupling permits the use of Sybase transactions at the PROLOG level, allowing for groups of rules to be specified as acting atomically, with the usual locking procedures on relations as are found in conventional database systems. This has the important advantage that one may integrate executable temporal logic programs with other database application programs in the usual manner for database systems, as possibility being followed by the TEMPORA project.

The restriction of using a bounded model in TEMPORA and the lack of support for periodic information can be recognized as deficiencies in the TEMPORA approach. Recently [KSW90], [BNW91] has suggested a method by which possibly infinite periodic temporal data can be stored and queried efficiently.

3 Temporal Logic: Syntax, Semantics and Execution

We shall restrict ourselves to considering the propositional linear version of USF. The principles which we present here are valid for predicate logics also (but by dealing only with the propositional version in this paper we ignore problems of quantification of variables). The general scenario in the executable temporal logic paradigm is that of a system programmed in temporal logic operating in conjunction with some environment. Thus in executable temporal logics, the atomic sentences are divided into two sorts: one sort whose members have their truth values controlled by the system, and another sort whose members have their truth values controlled by the environment. Hence in USF we have

- *action* propositions that are controlled by the system and, therefore, can be treated imperatively as well as declaratively,
- *environment* propositions that are not controlled by the system and may only be queried, i.e. their truth value can be checked, but the system cannot imperatively force their truth value during execution.

Definition 3.1 Propositional Linear Temporal Logic

We define the well-formed formulae (wff) of a propositional linear temporal logic USF by the following:

1. We have environment propositions from a set Σ_e , and a set of action propositions from a set Σ_a . The set $\Sigma = \Sigma_a \cup \Sigma_e$ is the set of all propositional variables in USF. $\Sigma_a \cap \Sigma_e = \emptyset$. Each member of Σ is a wff of USF.
2. \perp is a wff of USF.
3. If A and B are wffs of USF, then so are $\neg A$, $A \Rightarrow B$, $A \mathcal{U} B$ and $A \mathcal{S} B$.

This is the minimal syntactic definition of USF, for convenience we add further connectives, defined in terms of the minimal connectives.

$\top \stackrel{\text{def}}{=} \neg \perp$	$\diamond \varphi \stackrel{\text{def}}{=} \top \mathcal{U} \varphi$
$\varphi \wedge \psi \stackrel{\text{def}}{=} \neg(\varphi \Rightarrow \neg \psi)$	$\blacklozenge \varphi \stackrel{\text{def}}{=} \top \mathcal{S} \varphi$
$\varphi \vee \psi \stackrel{\text{def}}{=} \neg \varphi \Rightarrow \psi$	$\square \varphi \stackrel{\text{def}}{=} \neg \diamond \neg \varphi$
$\bigcirc \varphi \stackrel{\text{def}}{=} \perp \mathcal{U} \varphi$	$\blacksquare \varphi \stackrel{\text{def}}{=} \neg \blacklozenge \neg \varphi$
$\bullet \varphi \stackrel{\text{def}}{=} \perp \mathcal{S} \varphi$	

□

3.1 Declarative Semantics

The models for USF are of the form $\mathcal{M} = (\mathcal{F}_{\mathbf{N}}, V)$ where V is the assignment of truth values to the propositions for each state in the frame, $\mathcal{F}_{\mathbf{N}} = (\mathbf{N}, <)$. This is defined below.

Definition 3.2 Assignments for USF

The assignment of truth values to the propositions for each state in the frame \mathcal{F} is given by a function $V : S \mapsto 2^{\Sigma}$. □

Thus for USF, V is a function $\mathbf{N} \mapsto 2^{\Sigma}$. The truth values of formulae of USF in this model are defined for the minimal connectives in the usual way in Definition 3.3, with each formula φ being assigned a truth value for each state $s \in \mathbf{N}$, written as $\mathcal{M} \models_s \varphi$. A formula φ is said to be true in model \mathcal{M} (written $\mathcal{M} \models \varphi$) iff it is assigned truth for every state in \mathbf{N} .

Definition 3.3 Interpretation of USF in \mathcal{M}

$\mathcal{M} \models_t a$	iff	$a \in V(t)$ if a is a proposition
$\mathcal{M} \not\models_t \perp$		for all states $t \in \mathbf{N}$
$\mathcal{M} \models_t \neg \varphi$	iff	$\mathcal{M} \not\models_t \varphi$
$\mathcal{M} \models_t \varphi \rightarrow \psi$	iff	$\mathcal{M} \models_t \varphi$ implies $\mathcal{M} \models_t \psi$
$\mathcal{M} \models_t \varphi \mathcal{U} \psi$	iff	for some $s \in \mathbf{N}$ such that $t < s$, $\mathcal{M} \models_s \psi$ and for all $u \in \mathbf{N}$, $t < u < s$ implies $\mathcal{M} \models_u \varphi$
$\mathcal{M} \models_t \varphi \mathcal{S} \psi$	iff	for some $s \in \mathbf{N}$ such that $s < t$, $\mathcal{M} \models_s \psi$ and for all $u \in \mathbf{N}$, $s < u < t$ implies $\mathcal{M} \models_u \varphi$

□

For the remainder of this paper we shall restrict ourselves to consider only frames based on the natural numbers, i.e. $\mathcal{F}_{\mathbf{N}} = (\mathbf{N}, <)$.

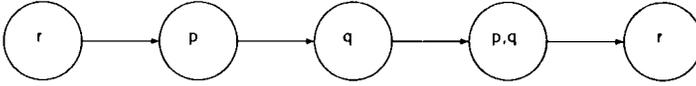


Figure 2: A Finite Temporal Model

Example 3.1

The (finite) model illustrated in Figure 2 is given by the following:

$\mathcal{M} = (\{1,2,3,4,5\}, <, V)$ where

$$V(t) = \begin{cases} \{r\} & \text{for } t = 1 \\ \{p\} & \text{for } t = 2 \\ \{q\} & \text{for } t = 3 \\ \{p, q\} & \text{for } t = 4 \\ \{r\} & \text{for } t = 5 \end{cases}$$

So for \mathcal{M} we have:

$$\begin{aligned} \mathcal{M} &\models_4 \blacklozenge p \wedge (p \vee q) \mathcal{S} r \\ \mathcal{M} &\not\models_4 \neg(\bullet q \vee \bigcirc r) \\ \mathcal{M} &\models r \vee \diamond r \\ \mathcal{M} &\not\models p \Rightarrow \bigcirc q \end{aligned}$$

□

3.2 Imperative Semantics

The ‘declarative past and imperative future’ paradigm is essentially about building the assignment V during the execution of the specification. Some of the values that V assigns, namely the values of the propositions in Σ_e , are determined by the environment. The remainder, i.e. the values of the propositions in Σ_a , are determined by the specification and the execution strategy.

During the execution of a temporal specification, the assignment V is built up so that at any given moment the database will have information about the past, the present and the projected future. For example, consider the following model at the moment just before time 3:

$\mathcal{M} = (\mathbf{N}, <, V)$ where

$$V(t) = \begin{cases} \{a, b\} & \text{for } t = 1 \\ \{b\} & \text{for } t = 2 \\ \{a\} & \text{for } t \in \{3,4,5,\dots\} \end{cases}$$

Note that this is just the representation of a historical database. Suppose that there is a single action rule specified for the database:

$$\bullet \neg a \wedge a \Rightarrow \bigcirc b \tag{1}$$

where $a \in \Sigma_e$ and $b \in \Sigma_a$. This rule can be read as ‘if a was not true at the previous moment of time but is true now, then b must be true in the next moment of time.’ If we wished to ensure the

truth of (1) at time 3 in the example model, we would have to decide whether $\bullet \neg a \wedge a$ was true at time 3, and if it was, ensure that $\circ b$ was also true at time 3. The atom a is clearly true at time 3, so the truth of the antecedent rests on the truth of $\bullet \neg a$, which is determined by the truth of $\neg a$ at time 2. So $\bullet \neg a$ is true at time 3 if $a \notin V(2)$. A look at our model shows that this is the case, therefore the antecedent of (1) is true, so we must ensure that the consequent $\circ b$ is true at time 3, therefore b must be true at time 4, i.e. $b \in V(4)$. Hence we would add more detail to V , so that the assignment V after evaluating the rule at time 3 would be:

$$V(t) = \begin{cases} \{a, b\} & \text{for } t = 1 \\ \{b\} & \text{for } t = 2 \\ \{a\} & \text{for } t = 3 \\ \{a, b\} & \text{for } t = 4 \\ \{a\} & \text{for } t = 5 \end{cases}$$

If the consequent of the rule had been $\circ\circ b$, we would have had to ensure that $b \in V(5)$; for $\circ\circ\circ b$, $b \in V(6)$ and so forth. Had the consequent been $\square b$, we would have needed $b \in V(t)$ for all $t > 3$.

The temporal connectives \circ and \square are *deterministic*; there is no choice about the changes to be made to V . Problems start when the consequents contain non-deterministic connectives such as \diamond and \mathcal{U} . If the consequent of our rule had been $\diamond b$, we would have had to ensure that $b \in V(t)$ for some $t > 3$. Which t should we choose? The specification gives no explicit help here, yet the choice can be important.

Example 3.2

Given the following specification:

$$\begin{aligned} a &\Rightarrow \diamond b \wedge \diamond c \\ b &\Rightarrow \neg \circ d \\ c &\Rightarrow \circ\circ d \end{aligned}$$

with a under the control of the environment, and b , c and d under the control of the system. Suppose that we are at time 0, with $V(0) = \{a\}$. Thus by the first rule of the specification, we have to make $\diamond b \wedge \diamond c$ true. Consider some of the possible ways this can be achieved:

- *make $\{b, c\} \subseteq V(1)$*
If we do this, when we evaluate the specification at time 1, the second rule will require us to make d false at 2, and the third rule will require d to be true at 3.
- *make $b \in V(1), c \in V(2)$*
Evaluating the specification at time 1, the second rule will require d to be false at time 2, and the third rule will require d to be true at 4.
- *make $b \in V(2), c \in V(1)$*
In this case, the second rule at time 2 will require d to be false at 3, and the third rule will require d to be true at 3. Thus we must not make this choice.

Of course, the environment can make a true again, at times 1,2,3,... which will complicate the above considerations. In any case, the execution must never allow the pattern of c being true followed by b in the next moment.

The problem can of course be circumvented by only executing deterministic specifications, and in many circumstances that is appropriate. Rules which express that something must eventually

take place are generally high-level statements of policy, or constraints on the behaviour of the system. In such cases, one may remove the non-deterministic rule from the specification and prove instead that it is a valid consequence of the specification. Nonetheless we may still wish to execute non-deterministic specifications: for prototyping purposes we may not want to have to determinise the rules, we may place bounds on the number of possible futures that may be described non-deterministically.

4 -Dynamic Behaviour of Historical Databases

In this section we present a small example of the dynamic behaviour of a historical database system, and outline some situations that may be encountered during the execution of temporal specifications which cause the declarative reading of the database and associated rules to be violated. The example database system involves a single action rule that, nevertheless, will allow us to illustrate several different situations. The example system is designed to perform the payment of invoices received by a company from its suppliers, upon appropriate authorization being entered into the database. The three predicates used in the database will be:

1. `invoice(l#, Amount)`: records an invoice which has arrived, identified by invoice number `l#` with `Amount` the sum to be paid.
2. `authorised(l#)`: records that the invoice `l#` is authorised to be paid.
3. `pay(l#, Amount)`: records the payment of `Amount` in response to invoice `l#`.

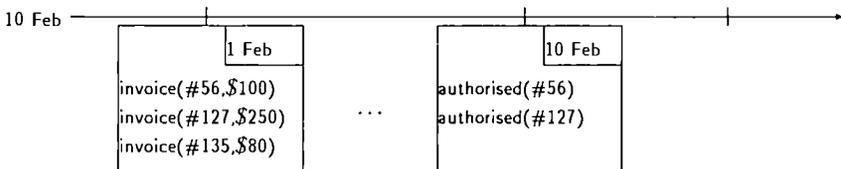
These predicates will hold at different instants of time, associated with the day of their occurrence. In the diagram below, we see the database state on the 9th Feb, in which the arrival of three invoices at the first day of the month has already been recorded.



The action rule specifies that if there exists an unpaid invoice in the database, and the unpaid invoice's payment is authorised, then the payment must be executed the next day. We can identify which invoices have not been paid since they arrived, by asking $\neg \text{pay}(l\#, \text{Amount}) \ \& \ \text{invoice}(l\#, \text{Amount})$. The action rule is then represented by:

$$\text{authorised}(l\#) \wedge (\neg \text{pay}(l\#, \text{Amount}) \ \& \ \text{invoice}(l\#, \text{Amount})) \Rightarrow \bigcirc \text{pay}(l\#, \text{Amount})$$

On 10th Feb two invoices are authorised to be paid, which we enter into the database to give:



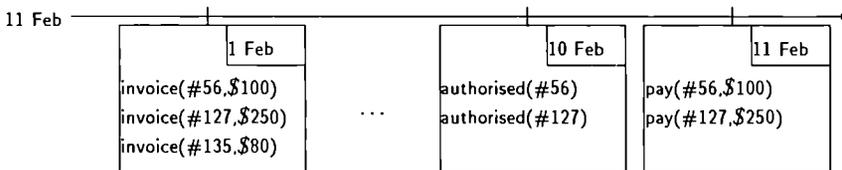
In this case, if we pose the following query to the database:

$$\text{authorised}(I\#) \wedge (\neg \text{pay}(I\#, \text{Amount}) \text{ } \mathcal{S} \text{ invoice}(I\#, \text{Amount}))$$

then we have the answer is the following:

$I\# = \#56, \text{Amount} = \100
 $I\# = \#127, \text{Amount} = \250

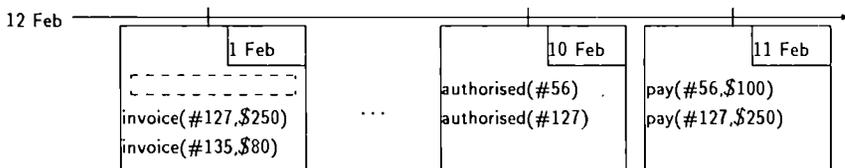
which implies that the antecedent of the action rule above now holds for two sets of values, and hence we must perform the actions $\bigcirc \text{pay}(\#56, \$100)$ and $\bigcirc \text{pay}(\#127, \$250)$. The state of the database after their execution is illustrated by:



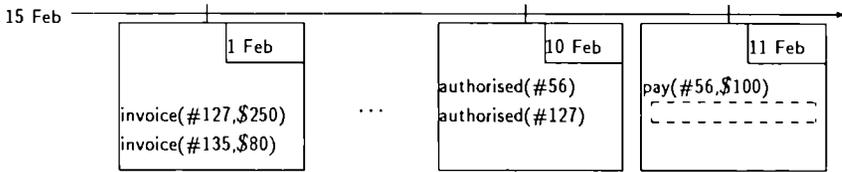
So far, what we have described just the normal operation of most information systems in which the *action detection* was followed by *action execution*. However it has been described in a declarative manner, since *after* the action has taken place we may still check the database to see that the rule holds (which it does since now both the query and action parts hold).

If we allow information about the past to be altered, we may encounter some problems with actions having been taken without any basis, or actions not having been taken which should have been. These problems are highlighted by the declarative reading of action rules, as the following three examples will illustrate.

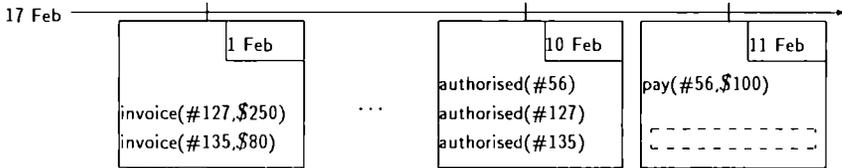
1. One day after the payment of the invoice #56, the original invoice is cancelled, which is represented by the deletion of invoice(#56,\$100) from the database. In this case, there is no longer any justification in the database for the payment executed on 11 Feb, and we say that pay(#56,\$100) has become a *non-supported action*. The database state is described below, where dashed boxes are used to indicate that there is some information 'missing' from the database.



2. On the 15 Feb, the information about the payment of invoice #127 is deleted from the database, perhaps reflecting the fact that the payment was declared void. The conditions for the payment action still hold in the database, but the action itself has been removed, and we will describe such a situation as an *action rule violation*. The database is illustrated by the following diagram:



- On the 17 Feb it comes to our knowledge that one week before there was actually an authorisation for the payment of invoice #135.

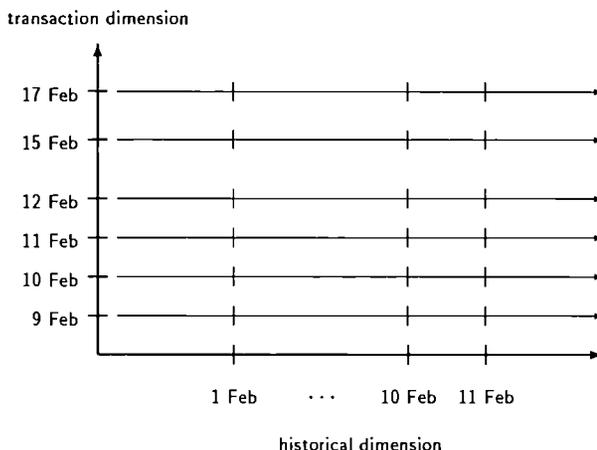


In this case, the new information is inserted in the database, changing the information about the past. We then realise that there should have been an invoice payment action on 11 Feb, $\text{pay}(\#135, \$80)$, but this action cannot be executed because it is impossible to execute an action in the past, and we call it an *action triggered in the past*.

The three examples show that no single course of action can be used to solve these types of problems. In (1) it would be sensible to simply note on some records that an invalid invoice has been paid, and we should try to get the payment made returned. In (2) the payment might have been deleted because it is realised that it was lost in the post, in which case a new payment is needed. In (3) we should execute the payment at the current time, since although late, it still needs to be made. As a general rule the course of action necessary to correct the inconsistency with the specification of the system behaviour is specific to the situation, and can only be decided upon after dialogue with the user.

4.1 Two Dimensional View

From the example in the previous section, it should be noted that we had several different views of the database at different times, i.e. the database is viewed differently on 12 Feb from 15 Feb and so on. We have therefore two notions of time, namely the historical time, which is associated with a piece of information in the database, and the transaction time, which is associated with the content of the whole database. This suggests a two-dimensional description on the evolution of the historical database. In the historical dimension we represent the contents of the database as viewed at one particular instant of time. In the transaction dimension, we represent the evolution of the database as seen from different moments of time. For the example presented above, the historical time associated with the database contents were 1, 10 and 11 Feb, while the transaction times in which the database was viewed were 9, 10, 11, 12, 15, and 17 Feb, as illustrated in the diagram below.



With a two-dimensional view of time we can describe not only how the information in the database evolves in the future, but also the way in which data about the past may be altered. In practice this allows us to monitor some special situations like the existence of *non-supported actions* and the *violation of an action rule*, both of them caused by changes in information recorded about the past in a database.

The way we detect those cases is as follows. Together with the execution of an action rule, we store in the database the *temporal dependencies* of the action: a list of time points where we found information giving support to the execution of the action. The temporal dependencies of an action are actually a compact form of storing relevant information about the state of the databases at previous transaction times. Whenever some information is changed about the past, the temporal dependencies are used to select the executed actions that must have their support re-evaluated. The full algorithm is described in [Fin90], and a prototype implementation produced as part of the SPEC research action. This process guarantees that *non-supported actions* and *action rule violation* are always detected, but the correction of their effects is a domain specific activity, and two main approaches can be taken. The system may produce a warning message telling the user that a special situation was detected, leaving to the user the responsibility of taking any appropriate action, or we may have *two-dimensional specifications* that are activated automatically when a special situation is detected, specifying what should be done. We note that the design of such domain-specific, corrective specifications is a non-trivial task of information engineering. For further discussion of these issues, see [Fin90].

Conclusion

It is well known that predicate calculus may be used for the querying of relational database, a practical implementation being DATALOG. However, for the querying of temporal databases, predicate calculus is not fully expressive, and can not describe the logic of update operations at all. We have shown that executable temporal logic provides a concise and complete formal system for the description of both query and updates to a historical relational database in a declarative manner. This has been used as the basis for an implementation by of executable temporal logic by the Université de Liège and Imperial College as part of the TEMPORA project, based on a coupling between BIM-PROLOG and Sybase also made as part of the project.

Executable temporal logic declaratively describes updating the future based on information about the present and the past. However, updating information in the past causes some anomalous situations in the database, that are handled by a two-dimensional view of the evolution of the historical database. This has been studied as part of the research action SPEC, and a sample truth maintenance system based on two-dimensional logic has been implemented.

Acknowledgements

We would like to thank all our partners on the TEMPORA and SPEC projects for their contributions to many discussions on the issues raised in this paper.

References

- [AM87] Martín Abadi and Zohar Manna. Temporal Logic Programming. In *Symposium on Logic Programming*, pages 4–16. IEEE, 1987.
- [AM90] Martín Abadi and Zohar Manna. Non-clausal Deduction in First-order Temporal Logic. *Journal of the ACM*, 37(2):279–317, 1990.
- [BNW91] M. Baudinet, M. Niezette, and P. Wolper. On the Representation on Infinite Temporal Data and Queries. In *Proceedings of the Tenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Denver, Colorado, 1991. ACM.
- [Fin90] Marcelo Finger. A Two Dimensional Approach to Historical Databases. Master's thesis, Department of Computing, Imperial College, 1990.
- [Gab87] Dov M. Gabbay. Modal and Temporal Logic Programming. In Antony P. Galton, editor, *Temporal Logics and their Applications*. Academic Press, 1987.
- [Gab89] Dov M. Gabbay. The Declarative Past and Imperative Future. In Behnam Banicqbal, Howard Barringer, and Amir Pnueli, editors, *Proceedings of Colloquium on Temporal Logic in Specification, Altrincham, 1987*, volume 398 of *LNCS*. Springer-Verlag, 1989.
- [GM91] D.M. Gabbay and P.J. McBrien. Temporal Logic & Historical Databases. In *Proceedings of the 17th International Conference on Very Large Databases*, Barcelona, 1991.
- [Hal87] Roger W. S. Hale. Temporal Logic Programming. In Antony P. Galton, editor, *Temporal Logics and their Applications*. Academic Press, 1987.
- [KSW90] F. Kabanza, J-M. Stevenne, and P. Wolper. Handling Infinite Temporal Data. In *Proceedings of the Ninth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Nashville, Tennessee, 1990. ACM.
- [Mos86] Ben Moszkowski. *Executing Temporal Logic Programs*. Cambridge University Press, 1986.
- [TC90] A. Tuzhilin and J. Clifford. A Temporal Relational Algebra as a Basis for Temporal Relational Completeness. In *Proceedings of the 16th International Conference on Very Large Databases*, Brisbane, 1990.
- [Ull88] J. D. Ullman. *Principles of Database and Knowledge-Base Systems*, volume 1. Computer Science Press, 1988.

GRAPHICAL INTERACTION IN A MULTIMODAL INTERFACE

H. BEN AMARA, B. PEROCHE,
Ecole des Mines, Saint-Etienne, France

H. CHAPPEL, M.D. WILSON,
Rutherford Appleton Laboratory, Chilton, UK

SUMMARY

The Multi-Modal Interface for Man Machine Interaction with Knowledge Based Systems (MMI2) project¹ is developing an advanced human computer interface to support co-operative dialogue between users and knowledge based systems. The interface supports natural language, command language, graphical display, direct manipulation and gesture. Several tools are incorporated to allow the graphical visualisation of the state of the knowledge base, its suggested designs for local area computer networks, and interactive charts and graphs. The mechanism for designing and presenting these visualisations which also supports the interaction of the graphical mode with other modes is described in this paper.

1. INTRODUCTION TO THE MMI2 SYSTEM

Human-human dialogues allow both participants in a conversation to change topic and take control of the dialogue using rules which guide the transfer of control and make it apparent who is in control at any time. Human-human consultations also have a discernible structure that is not a simple sequence of questions and answers, but includes openings and closings to subdialogues and changes of topic. In such conversations both parties have a view of this developing structure and co-operate to move it to a conclusion. Further, to support many technical human conversations verbal language is used, not alone but in conjunction with graphical representations appropriate to the topic of the conversation which are pointed at and referred to by both parties (for example, when discussing a route across London a map of underground railway stations may be used which represents the links between them).

The objective of the Multi-Modal Interface for Man Machine Interaction with Knowledge Based Systems (MMI2) project is to develop a demonstrator of a highly interactive interface for Knowledge Based Systems (KBS) which will facilitate co-operative dialogue using mixed graphical and natural language modes and which will allow KBS to be developed that support the richer aspects of human dialogue rather than the conventional sequential question and answer structure of expert systems. In the demonstrator the KBS guides users through the design of computer networks, making suggestions and criticisms as though it were a well informed consultant. The MMI2 interface allows the user and the system to ask questions and make replies in natural language (in English, French or Spanish), a command language or through graphical interaction using direct manipulation or gesture. Users are able to interrupt the system by asking

¹ The MMI2 project has seven European consortium members. The partners are: BIM (Prime Contractor), ADR/CRISS, Ecole des Mines de Saint-Etienne, INRIA, ISS, Rutherford Appleton Laboratory, University of Leeds

questions of the system instead of merely replying to questions. The system will be able to justify its reasoning and elaborate on those justifications if required (see (24 and 23) for reviews of similar systems).

Users can input a building structure diagrammatically; locate specific items of computer network equipment on the plan by manipulating icons; specify requirements for network load and usage in natural language; ask the system to complete the design which will be presented graphically; enquire about the design and have the answers presented as charts or tables; or enquire about the reasoning process behind the design and receive natural language replies tailored to the system's model of their knowledge and goals. In such applications a tight link is required between the graphical and textual interactions, and these must be mapped onto an underlying KBS which produces replies and explanations. The screen appearance of the MMI2 system is shown in Figure 1.

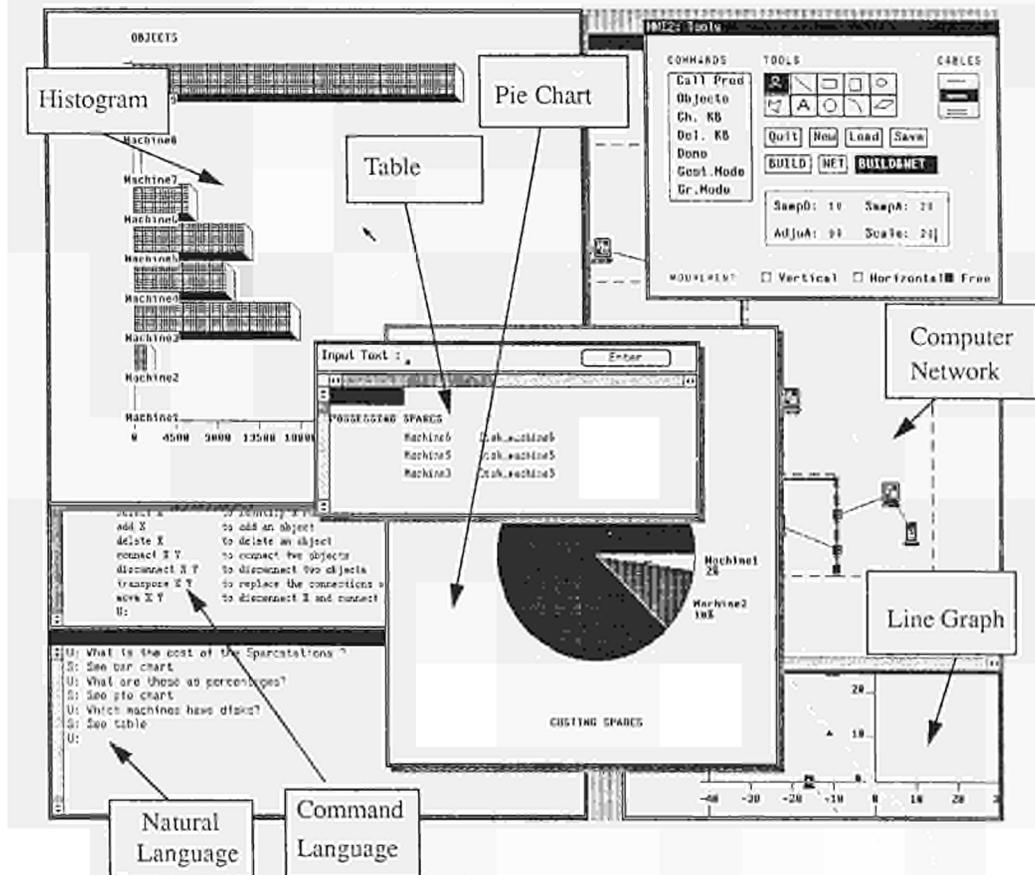


Figure 1: Typical Screen Layout of the MMI2 system in use

In order to support multi-modal communication the architecture of the MMI2 system is divided into three layers: the bottom layer represents the application KBS; the middle layer manages the dialogue between the user and the system; and the top layer incorporates the individual modes. The overall architecture of the MMI2 system has

been described elsewhere (8) and will not be reviewed here. This paper will focus on the mode layer and refer to the Dialogue Controller (DC) and the User Modelling module (UM) in the dialogue management layer.

The processes required for dialogue management, as with the modes themselves are allocated to modular experts in the overall MMI2 architecture. The interaction of these experts is governed by a coordinating Dialogue Controller, and the communication between them uses a common language to represent the semantics required for all modes, which is termed the Common Meaning Representation (CMR). This language is required to be a maximally expressive formalism for the internal representation of the semantic content of dialogue utterances in whatever mode. It is a typed first order logic using promiscuous reification of events and relations (after 13). The interaction between the dialogue management components and the underlying application KBS uses an application specific language which would be specified for each application.

The DC receives packets of CMR from the input modes and acts on them by drawing on an existing model of the task, communication plans and the application KBS. It then produces packets of CMR which are sent back to the mode layer for presentation to the user. To produce a relevant explanation tailored to the user, the DC and the modes must have knowledge of the user's goals, beliefs and preferences which are represented in the UM.

The content of the User Model and other domain dependent parts of the interface were derived by Wizard of Oz studies (see 12) of potential users and through the knowledge acquisition for the underlying knowledge based system. When the project is completed, not only is a toolkit to be produced so that the interface may be ported to other domains, but also a development method incorporating these techniques for acquiring the required domain knowledge.

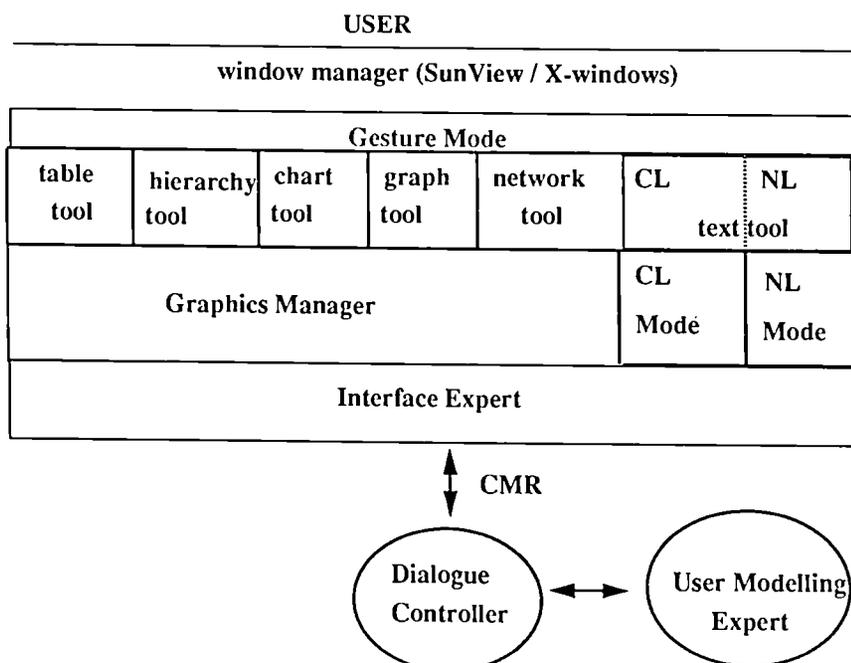


Figure 2: The architecture of the MMI2 mode layer

2. THE MODE LAYER IN MMI2

The mode layer of the MMI2 is illustrated in Figure 2. This shows that the DC communicates with the Interface Expert (IE) through CMR packets. The role of the IE here is simply to provide a buffer to store system output and deal it out to the relevant mode when it is available; or to store user input until the DC is ready to read it. User input is received by the IE in the form of CMR packets from the Command Language mode (CL), Natural Language mode (NL), Gesture mode or graphics mode through the Graphics Manager. System output can be passed to either the NL or graphics modes, the CL and Gesture modes are not used for system output.

The CL and NL modes both translate user input into CMR. The NL mode also translates CMR into natural language. In both cases the user directly interacts with a simple text editor style input tool which stores the user's input until a carriage return is pressed before passing it to the mode. Similarly, the tool formats system output and presents it once the NL mode has produced natural language strings.

Graphical interaction is handled in two layers. The outer layer consists of a series of interactive tools which visualise information using different conventions. Each of these individual tools has its own data structures which must be created from the CMR by the Graphics Manager layer below them.

The Gesture mode operates by catching the events from the mouse on the windows of any of the graphical tools when the user has selected gesture as the mode to use from the menu on a tool. The gesture mode recognises hand drawn gestures (such as editing symbols) by correlating the pattern of features in the event stream with a pre-stored library of up to 50 gestures for each application. Neither a larger library, nor a neural net approach (see (21)) are used since gesture is not the only mode of interaction and is only used when it is most convenient. Studies of potential users have shown that this set size is sufficient for the applications investigated. Once a gesture is recognised, the arguments to the gesture (e.g. MOVE with arguments Machine1 and Room2) are requested from the tool on which the gesture was drawn using the screen co-ordinates of each argument and the semantic selection restrictions on the argument as cues.

The Graphical components of the mode layer will now be described in more detail in the remainder of the paper.

3. GRAPHICS TOOLS AND THE GRAPHICS MANAGER

The graphics manager translates CMR into the data structures required by each of the graphics tools, and translates the output from the graphics tools produced when the user acts on them into CMR. The DC decides which mode to use to present information to the user on the basis of the effectiveness and efficiency of a mode to convey the information or request. When the graphics mode is chosen, the graphics manager must then decide which graphics tool to use to visualise the information and convert the logical CMR representation into that required for the chosen tool. The tools so far available allow information to be presented as building or network diagrams, as pie charts, histograms, tables, line graphs or scatterplots, or as hierarchies. These visualisation tools all allow the presentation of information but also allow the user to directly manipulate the representations; for example, users can move the values shown on bar charts to produce CMR packets which are sent to the DC to alter the state of the KBS.

The next section describes in detail the internal structure of the tool for displaying three dimensional models of buildings superimposed with network diagrams. The facilities provided by the graphics manager to support the interaction of graphical and natural language modes are described here.

The core set of operations which can be performed on the graphical tools are to select, move, add and delete objects. These operations can be performed by the user on the graphics tools and by the graphics manager on a tool. The tool which displays computer networks also allows the objects to be linked together, have links removed, and the identity of all the objects, or of the object at a specific location queried by the system. The graph and table tools also allow the values of objects to be set by the user and the system. This set of operations is sufficient to allow the logical CMR from the DC to be converted into actions on the tools by the Graphics Manager, and for the user actions to be recoded as a series of terms in CMR formulae. The variables used for objects in the graphics tools are different from those used in the CMR so a table is maintained in the Graphics Manager to map between these.

Users can perform actions in each tool which: select menus; are interpreted as gestures; act to change the appearance of the tool used (e.g. changing the fillings of graph bars or other presentation features); or which are communication actions that must be encoded in CMR for the dialogue management system. Actions which are not communication actions do not result in CMR since they are not communicating with the KBS. However, they do result in changes to the UM which stores such preferences. In these cases direct calls are made to the UM to store that a preference for a particular presentation style is preferred by that user for that class of visualisation within the task being performed. Whenever a graphical tool presents information the Graphics Manager always reads the UM to determine the relevant preferences, which will either be those set by the user explicitly, or those inherited from a stereotype to which the user belongs.

Communication actions include the selection of an item as a referent for a natural or command language operation, or the explicit creation or adjustment of information presented; for example, a user may type the natural language command "move [select] this workstation [select] here." or the request "What is the disk capacity of [multiple selection] these". When users make communication actions the graphics manager will construct a CMR representation for the action and pass it to the DC, this may result in the DC returning a CMR packet to be displayed. Similarly, users can ask for information in another mode which may result in a CMR packet being sent to the Graphics Manager for graphical presentation. In either case, when the Graphics Manager receives a CMR packet from the DC it will decide if it can be displayed as network diagram, hierarchy or graph (it is intended to also include tools to present set relationships using Venn diagrams although this is not yet implemented). The Graphics Manager will then dynamically design the data structure for the chosen visualisation tool, and then call that tool to render the image. If none of these, it determines if it can be represented as a change in the state of an existing representation. If it is unable to determine a way of presenting the information it will return the CMR packet to the DC with amendments as to why it failed to present it. This will result in the DC choosing another mode to present the information. The details of the design method are too lengthy to describe here, although they are available from the authors.

In addition to translating user actions on the graphical tools into CMR and presenting CMR through the graphical tools, the third role of the graphics manager is to support the use of graphical referents and descriptions in natural language. Each of these forms of reference is ultimately resolved by the processes used to resolve reference in natural

language. Three classes of support are offered which enable the interaction of modes by constraining the set of possible referents in the natural language component.

Firstly, non-ostensive deixis - the indication of a natural language referent in the context of the utterance without pointing, e.g. "move the fileserver" where there may be two but only one is shown on the screen (this would be anaphoric reference if it were not visible on the screen but had been mentioned in the dialogue). Non-ostensive deixis is resolved in MMI2 by the Graphics Expert constantly updating a list of possible discourse referents which are visible on the screen in CMR terms. This is used by the DC to disambiguate whether a textual referent should be chosen from its list of previously mentioned textual referents (anaphoric referents), or the current list of possible non-ostensive deictic referents. The DC has direct access to this list at all times.

Secondly, reference in natural language to objects by their graphical features. The resolution of referents from definite descriptions of domain objects by their graphical features requires that the graphical features of objects be available to the natural language referent resolution mechanism as well as the real world features. For example, in a discourse about the performance of local area network which is displayed diagrammatically, one workstation is shown as a purple icon. A user intending to remove this machine from the network could type any of:

- 1) *remove the Stellar workstation*
- 2) *remove the purple workstation*
- 3) *remove the purple icon.*

All of these use natural language descriptions of the intended item. The first uses a domain description, and the third uses a graphical description. The second is ambiguous as to whether the workstation in the world is purple or the icon representing the workstation is purple. All of these require processes which identify linguistic referents to operate, but the second two require that those processes have access to descriptions of the graphical representation of the domain objects.

Thirdly, natural language references to graphical spatial relationships. This is an extension of the previous problem where the definite reference is not merely in terms of the graphical description of the object but includes spatial relationships in the graphical display. In the same example of a displayed network, the user could ask to "remove the top workstation" or "remove the left workstation". In both these cases the spatial relation is ambiguously one in the domain or in the graphical representation. For example, "the top workstation" could be that on the uppermost floor of the building or that on the upper edge of the displayed diagram. This ambiguity must be resolved by the processes which identify linguistic referents using the context of the utterance, or by asking the user to disambiguate the example explicitly. However, in order to appreciate the ambiguity these processes must have access to a representation of the spatial geometry of the domain and of the user's perception of the machine representation. It is intended to address both of these issues in the MMI2 project although no resolution is currently available. Both problems could be overcome if users identified referents by selecting graphical representations with the mouse, but this may not be the preferred method. As with other aspects of "naturalness" in the interface, this will require studies of users performance preferences as these systems develop. A further description of these issues is given in (27).

4. THE NETWORK TOOL

In this section, we will present the interactive keyboard tool for the input of building maps and of network diagrams. The main aims of the Network Diagram are:

To allow users to draw buildings and networks either by free hand drawing or by means of graphical tools described below. In this first version, we have decided to restrict the drawing to only one building with only one floor. The user can draw separately a building and a network which will be represented by two distinct planar maps. However, we must establish some links between buildings and networks: for example, a machine is inside one room, a cable skirts around walls, etc.... For this purpose, we have developed some functions which attach attributes to the data structures.

To make the pre-processing of the drawing (sampling, adjusting, erasure elimination) To deduce the semantics of drawings from the planar map data structures.

To develop some functions allowing communication between Network Diagram and the other modules (for example: to move a machine from one room to another, using either natural language or command language or gesture).

4.1. THE GRAPHICAL INTERFACE

The Network Diagram interface contains three windows (see figure 3):

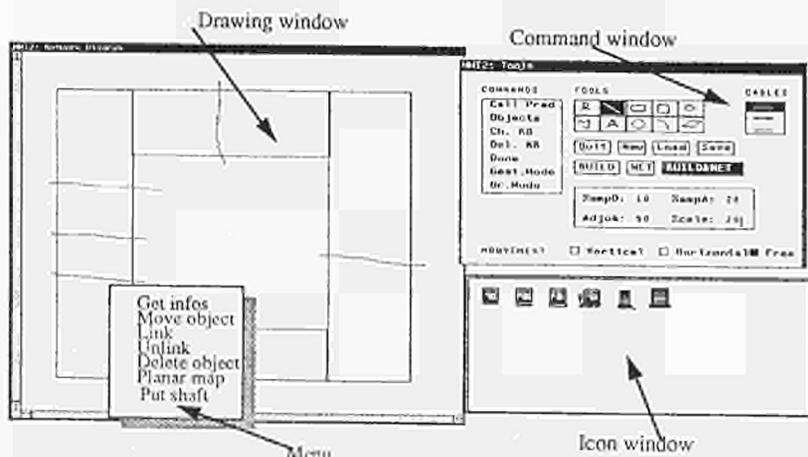


Figure 3: The graphic interface

A drawing window, where we can introduce building plans and install networks. In this window, we can display either a building or a network or both. Insertions of line-drawings may be done with a mouse in the same way one would use pencil and rubber. However, some tools are provided to help the user to introduce plans.

A command window, with a command menu, a cable menu, buttons, drawing tools and parameters. Some items of the command menu are used to communicate directly with the application; the two last items allow user to switch between graphical and gestural modes. The first range of buttons contains useful commands (Quit, New,...), and the second one allows user to display either a building or a network or both. The cable menu is used to specify the cable type. The user may modify textually the default values of Sampling and Adjusting parameters (which will be defined in section 4.5.2.1).

An icon window, which is displayed when the user makes the input of the network, contains some icons representing computers, disks, taps and terminators.

4.2. THE USE OF THIS INTERFACE

The input of building and network plans is made in two stages.

1) Input of building plans

The user draws the building map with the mouse (see figure 4). When the unrefined drawing is judged satisfactory enough, the user starts actually the planar map construction by selecting in the menu the item "planar map". The building map is returned in an embellished form where line-segments have been sampled, adjusted and erasures eliminated (see figure 5). The sampling and adjusting parameters (samplD, samplA, adjusA) may have been previously updated by the user. The line-segments obtained in this way are interpreted as being walls; in the same way, the faces of the map are assumed to be rooms. The system incrementally attaches identifiers to these rooms and walls during the planar map construction. After the planar map has been constructed, some operations on the objects are available:

- deletion of one or several walls. First, the walls to be deleted have to be selected; these walls are then highlighted. The deletion command can now be activated from the menu. This deletion automatically involves the reconstruction of the planar map. In fact, an improvement of the current version will be done to allow incremental updating of the planar map data structure (cf section 4.4.2):
- walls insertion: this operation acts in the same way than the previous one. However, planar map reconstruction is done only on the user's request.
- technical shaft installation: if technical shafts have to be installed along the walls, the user must select these walls and activate the "*put shaft*" item. The shafts are then displayed using a special texture (see figure 6).

Displaying information: the user can get information about an object by activating the "*get info*" item.

For convenience, rooms identifiers are displayed in the bottom left corner of each room (see figure 6).

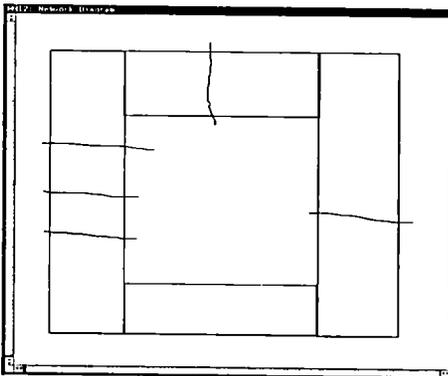


Figure 4: the unrefined drawing

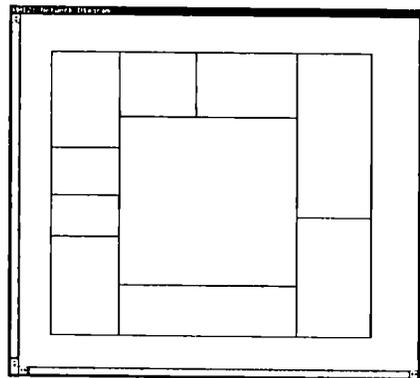


Figure 5: the drawing after the construction of the planar map

2) Input of network plans

There are two ways to make the input of the plan of the network: in the first one (see figure 6), the user draws the plans of the network on a window in the background of which the plans of the building are displayed with dotted lines (BUILD NET button). This allows the user to put machines and cables according to the location of rooms and walls. In the second one, the user draws the network plans on an empty window (NET button).

In fact, buildings and networks are represented by two distinct planar maps and the graphic tool should provide automatically links between these two data structures, so the location of network objects is important. That is the reason for which the first way we suggest is preferable. The second way is useful when the user has to modify a known network.

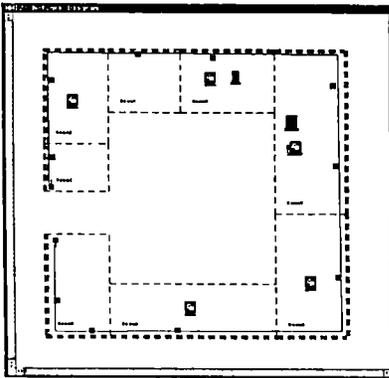


Figure 6: network objects

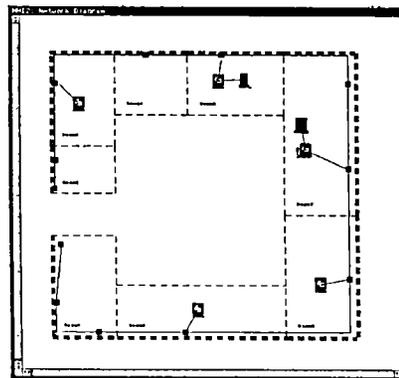


Figure 7: construction of the planar map

Let us now explain the use of this tool. First, the user installs cables which must be inside the building; then, he puts objects like computers, disks, taps, terminators... These objects must also be inside rooms (see figure 6). Let us notice that taps and terminators locations must be as near as possible of cable locations. Indeed, when the system constructs the planar map, it adjusts tap and terminator locations by projecting their coordinates on the nearest cable. Now, the user may connect computers and disks, computers and taps. For these purpose, he must select the two objects to be connected and activate the "link" item, which creates a physical link between them. Finally, the user constructs the network planar map (see figure 7). Several available operations on network objects are described below: moving of an object; displaying information; deletion of an object; modification of the cable type; connection between two objects; disconnection of two objects. When the user is satisfied, he must inform the rest of the system about the end of the operations by activating the "Done" item. At this point the data structure is read and passed to the Graphics Manager which translates it into CMR.

4.3. THE DESIGN OF THE NETWORK TOOL

The objectives of this tool were: to depict the topology of buildings and networks; to depict the geometry of buildings and networks; to give to the user a freehand tool for

graphic input and manipulations; to allow the deduction of the semantics of the input. For the realization of this tool, three problems were to be solved:

1) What data structures to choose in order to represent both topology and geometry of the drawing?

Among solutions which have been taken into account, there was the dynamic data structure describing plane configurations studied in (20) and the extended quadtree as defined in (3). These data structures seemed to be interesting, but they appeared to be less good than that of planar map which we define here.

If X is a set of points and E a subset of $X \times X$, a graph $G = (X, E)$ is said to be planar if it has a representation in the plane such that two distinct edges intersect only on a point. A planar map is a plane representation of a planar graph. Such a data structure is made up of two parts: a local planar map, inspired by the Winged-Edge Structure from (4) or the Doubly Connected Edge List from (19), which represents the topology of the drawing and a global planar map, which represents its geometry. This data structure will be described in detail in section 4.4.

2) A freehand input leads to erasures or malformations.

It is so essential to expect a process allowing to delete the maximum of erasures and burrs before the construction of the planar map. Furthermore, from a user point of view, it is pleasant to be returned a well built drawing without taking care over his own drawing.

3) Semantic deduction.

The general solution to this problem has been described in the previous section.

4.4. PLANAR MAPS

Much work has been done on planar maps, studying methods for combining them (10, 16, 9, 25, 17]) and applying them (11). These studies focus on two facts: firstly, dividing the topological and geometrical sides of a drawing; secondly, the consistency of these two sides. The data structure is generic and not limited to this application.

The construction of a planar map requires the computation of points of interaction between edges. Our first implementation used Bentley-Ottmann algorithm to compute the planar map. This method was not incremental and the map had to be recomputed each time a new segment was added. We have also studied a method to build a data structure to support incremental insertion or deletion of segments in a planar map, dynamically computing new intersections and updating the data structure.

In this section, we will first present some definitions of planar maps. Then, we will present a non-incremental method to construct a planar map where we will outline the Bentley-Ottmann algorithm and some problems linked to numerical accuracy which we encountered. Finally, we will discuss a method to incrementally build a planar map.

4.4.1. SOME DEFINITIONS

- *Map and dart*

A map is a 3-uple, $G = (B, \sigma, \alpha)$ where

- B is a finite, non empty set of darts; a dart will be graphically represented either by an oriented edge or by an half-edge. Notice that, intuitively, a dart defines a vertex.

- σ is an involution on B without fixed points: $\forall b \in B, \sigma^2(b) = b$ and $\sigma(b) \neq b$

- α is a permutation on B .

Nota bene: we will denote $A\tau^*$ the orbit $\{\tau^p(b) / b \in A, p \geq 0, A \subseteq B\}$. It is the set of darts included in B attainable from those of A with the help of τ .

- Edge and vertex

Let $G = (B, \sigma, \alpha)$ be a map.

$a = b \sigma^* = \{b, \sigma(b)\}$ where $b \in B$ is called an edge of G . Darts b and $\sigma(b)$ are called opposite.

$s = b a^* = \{b, \alpha(b), \alpha^2(b), \dots, \alpha^{k-1}(b)\}$, where k is the smallest integer such that $\alpha^k(b) = b$, is called a vertex of G . Darts b and $\alpha(b)$ are said to be adjacent.

- Boundary

We call **direct boundary** (resp. **backward boundary**) the set $\{b_{i_1}, b_{i_2}, \dots, b_{i_n}\}$ from G where

$$\forall k, \exists j: b_{i_k} = \alpha^{-1} \circ \sigma(b_{i_j}) \text{ (resp. } \alpha \circ \sigma(b_{i_j}) \text{)}.$$

In the following of this report, the term boundary will name a direct boundary.

A boundary traversed clockwise is said to be **outer**; otherwise, it is said **inner**.

- x-order

The lexicographical order of points in the plane will be called **x-order**. This order is total (see figure 8).

- y-order

Let us consider a sweep-line, parallel to the y-axis. Edges crossed by this sweep-line are said to be **active**. The set of active edges can be totally ordered by increasing ordinate of their point of intersection with the sweep-line. This order will be called **y-order**. Of course, both the y-order and the set of active edges depend on the abscissa of the sweep-line (see figure 11).

- α -order

A total order, called **α -order**, may be defined on the darts incident to a vertex x : the half straight line which bears any dart makes an angle α ($0 \leq \alpha \leq 2\pi$) with the vertical half straight line coming down from x . Darts are ordered according to angle α (see figure 9).

- Birth and death edges

An edge is born at its smallest vertex according to the x-order; it dies at its greatest one.

- Birth, life and death vertices

A vertex S is said to be a **birth** (resp. a **death**) vertex if all the edges converging to this vertex are birth (resp. death) edges (see figure 10 (a)). It is said to be a **life vertex** if there are both edges which are born or which die in this vertex (see figure 10(b)).

- Dangling, boundary, connecting, final, isolated edges

.An edge $a = \{b, \sigma(b)\}$ is said to be **dangling** if darts b and $\sigma(b)$ belong to the same boundary (see figure 11). It is said to be a **boundary** one if it is not dangling.

. An edge $a = \{b, \sigma(b)\}$ is said to be a **connecting** one if it is dangling, if $\sigma(b) \neq b$ and if $\alpha(\sigma(b)) \neq \alpha(b)$ (see figure 4).

. An edge $a = \{b, \sigma(b)\}$ is a **final** one if it is dangling, if $\alpha(b) \neq b$ and if $(\alpha(\sigma(b)) = \alpha(b))$ or $((b) = b$ and $\alpha(\sigma(b)) \neq \alpha(b))$ (see figure 11) and **isolated** if it is dangling, if $\alpha(b) = b$ and if $\sigma(b) = b$ (see figure 11).

4.4.2. NON-INCREMENTAL CONSTRUCTION OF THE PLANAR MAP DATA STRUCTURE

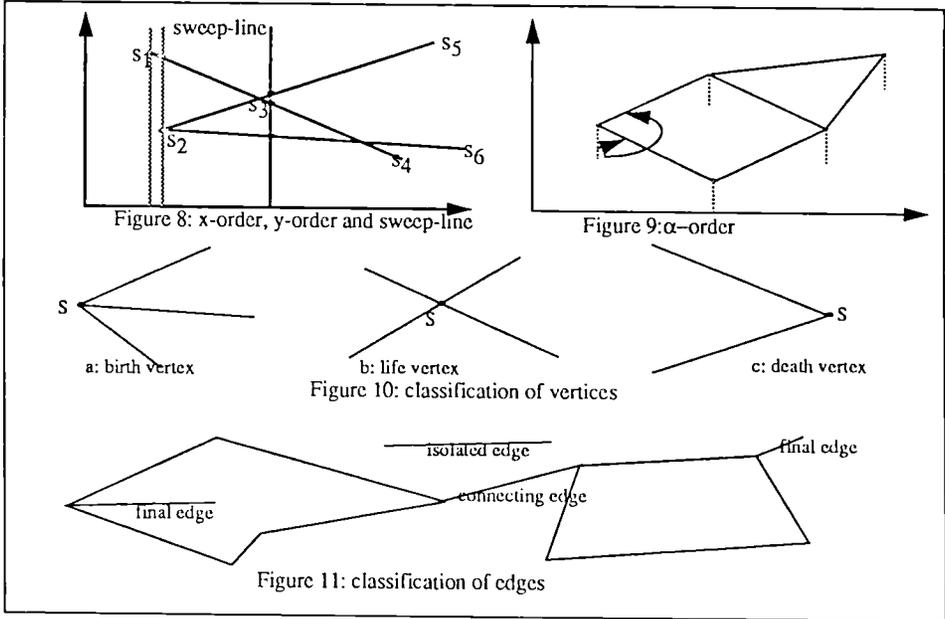
The planar map data structure is built in three stages: initialization, construction of the local-PM and construction of the global-PM.

4.4.2.1. INITIALIZATION OF THE PLANAR MAP DATA STRUCTURE

The first time, we build a data structure associated to each vertex by traversing the list of initial points. The data structure for vertices is a doubly-connected list where the vertices are totally ordered according to the x-order. Between two consecutive vertices, another data structure is built to represent the edge having these vertices for ends. Each edge contains a 2-dimensional array representing the two darts of the edge.

In this stage, we don't take into account possible intersections between edges. So, in a second stage, we will have to find intersections between edges. During the construction of the planar map data structure, we have admitted to join vertices which lie inside a circle with given radius. Thus, a vertex will be created only once.

Each dart points to the next dart according to the α -order. Furthermore, the last dart of a vertex points to the first one of the same vertex. The α -order is so defined by a circular list of darts. The number of edges incident to a vertex is not limited, although finite of course. Furthermore, it is possible to label vertices, edges, darts and boundaries with informations linked to the aimed application (for example, a color, a texture, a name...).



4.4.2.2. CONSTRUCTION OF THE LOCAL-PM

From the data structure obtained during the initialization stage, the construction of the local-PM consists in the computation of intersection points between initial edges and the updating of the data structure, by adding new vertices associated to the points of intersection and by replacing each edge holding a point of intersection by two new edges.

During this stage, some kinds of erasures may be removed. For example, two edges with the same birth (resp. death) vertex and with the same direction can be merged in only one edge. To find points of intersection between N edges, the simplest method consists in checking the intersections between all the pairs of edges. But this method is not impressive: its complexity is $O(N^2)$. So, we chose to use Bentley-Ottmann algorithm (6), which has a better complexity.

This algorithm uses a vertical sweep-line, moving from left to right. Its principles are based on the following two remarks: first, the only useful sweep-lines are those which go through a vertex or a point of intersection already known. Modifications for y-order are made by removing death edges and by inserting birth edges, which become active. Secondly, two edges may intersect if and only if there exists a position of the sweep-line for which they are consecutive.

For the complexity analysis of Bentley-Ottmann algorithm, let us call N the number of initial edges and K the number of intersections. In the worst case, $K = \frac{N(N-1)}{2} = O(N^2)$. In this case, Bentley-Ottmann algorithm has a running time $O(N^2)$, like the naive algorithm. Otherwise, the running time of the algorithm is $O((N+K) \log N)$.

We must notice that some problems arise when this algorithm runs, due to numerical inaccuracy inherent to floating point arithmetic. Unlike other algorithms where numerical errors remain local, there is propagation of these errors in Bentley-Ottmann algorithm. Michelucci studied the effect of numerical inaccuracy on some methods for the computation of points of intersection between two edges. He showed that consequences are more serious for Bentley-Ottmann algorithm.

Numerical inaccuracy has serious effects not only on the computation of points of intersection, but also on the orientation of darts around a vertex. This orientation is based on angle comparisons between two edges with a common vertex. The angular uncertainty becomes significant when the length of the edges is small. The effects of numerical inaccuracy may produce fatal errors, which make `local_PM` incoherent with reality. Several approaches have been proposed to solve this problem (22, 15, 26, 18).

4.4.2.3. CONSTRUCTION OF THE GLOBAL_PM

At this stage of the algorithm, all the points of intersection have been found. Now, the algorithm must create boundaries, insert them in the inclusion tree and mark on each dart the boundary to which it belongs. To determine a boundary, it is sufficient to be able to find a dart belonging to this boundary. Each boundary is labelled with its first dart: it is the one having the smallest vertex according to the x -order. If several darts satisfy the previous condition, the chosen dart will be the smallest one according to the a -order. This dart will be called the birth dart of the boundary.

We also have to represent the natural order induced by the inclusions between boundaries. This may be done by a binary family tree: each boundary points to its father, its son and its youngest brother. A total order is defined among brother boundaries, with the help of their birth vertices. When two such boundaries have the same birth vertex, they are ordered according to the a -order. An outer boundary is obtained by traversing the `local_PM` clockwise and in inner one by an anticlockwise traversal.

For convenience, a virtual inner boundary, called infinite boundary is created, without dart, which is represented by the root of the inclusion tree.

Let us notice that an inner boundary is necessarily contained in an outer one but the infinite boundary; conversely an outer boundary is necessarily contained in an inner one.

Another data has been introduced: the stop edge of a vertex, which is used to locate rapidly an edge or a vertex. For a birth vertex, the stop edge is the first low_active edge for the y -order which is not born at that vertex. In the other cases, the stop edge is the first edge, for the a -order, dying at this vertex.

The algorithm to construct the `global_PM` from the `local_PM` can be summarized by the following pseudo-code:

Construction of the global_PM

```

{
  for each vertex S taken according to x-order do
    for each edge e arising at S according to  $\alpha$ -order do
      if (the boundary of the dart 2 of e does not exist) then
        {
          create boundary B;
          traverse darts of B and mark them as borrowed by B;
        }

```

```

affect birth dart of B;
if (S is a birth vertex) and (e is the first edge incident at S)
then {
  B is outer;
  if (the stop edge of S does not exist) then
    B' ← virtual boundary of the map;
  else B' ← boundary of dart 1 of the stop edge of S;
  if (B' is inner) then the father of B is B';
  else the father of B is the father of B';
}
else {
  B is inner;
  B' ← boundary of dart 2 of the birth edge of B;
  if (B' is outer) then the father of B is B';
  else the father of B is the father of B';
}
add B at the end of the list of the sons of the father of B;
}
}

```

4.4.3. INCREMENTAL CONSTRUCTION OF A PLANAR MAP

As mentioned before, the problem addressed here is to allow incremental insertion or deletion of new segments in a planar map, to dynamically compute new intersections and to update the data structure. In this case, topological information has to be deduced from geometrical one. When two segments intersect at a new vertex, the ordering of the fourth edge around the vertex provides topological information used to follow the contour of the face incident to the vertex. Since exact arithmetic is used in this process, the map topology is always consistent with the geometry of the drawing.

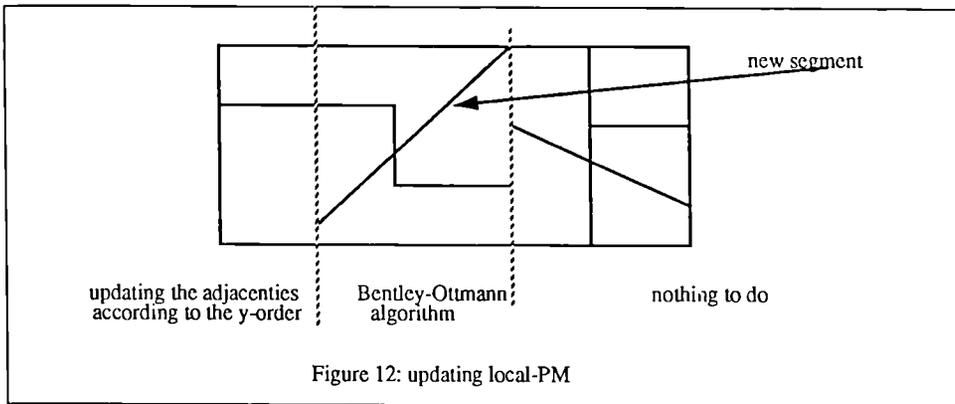
The incremental manipulation of the planar map is made by two stages: updating the local-PM and updating the global-PM. In this process, only two operations are allowed on the map: edge erasing and edge insertion.

4.4.3.1. UPDATING LOCAL-PM

During the planar map construction process, a new segment is intersected with a subset of the segments already inserted in the map. Assume we have a map already constructed and we want to add a new segment; intuitively the algorithm of insertion is as follows (see figure 12):

- the two end vertices of the segment are inserted in the x-order list of vertices;
- we go all over the vertices and update the adjacency stack of edges until we find the first vertex of the new segment;
- we run Bentley-Ottmann algorithm and update the map by inserting the new intersection vertices and ordering the new edges around vertices. The insertion algorithm is stopped when the last vertex of the new segment is reached.

Two special cases must be handled: rounded segments with null length and partially overlapping lines. Some processes are provided to resolve these kinds of problem.



4.4.3.2. UPDATING THE GLOBAL_PM

This algorithm is inspired by Gangnet algorithm (11). It classifies the edges as following:

- 1- a final edge with both sides into an inner boundary.
- 2- a connection edge with both sides into an inner boundary.
- 3- a boundary edge with both sides in two distinct inner boundaries.
- 4- a boundary edge with one side into an inner boundary and the other side into an outer boundary.
- 5- an isolated edge.
- 6- a final edge with both sides into an outer boundary.
- 7- a connecting edge with both sides into an outer boundary.

The addition of an edge implies the creation its darts, its vertices and the updating of the α -order. The inclusion tree is updated after each new addition; it is therefore possible to find the type of a new edge by counting its handling vertices and checking the updated boundary. The inclusion tree is then updated by performing the following actions:

```
switch (edge type)
{
  case 2: one inner boundary and one outer boundary are merged to give a single inner boundary
  case 3: one inner boundary is split to give two inner boundaries
  case 4: one outer boundary is split to give an outer boundary and an inner boundary
  case 5: an outer boundary is created
  case 7: one outer boundary and one inner boundary are merged to give a single outer boundary
}
```

When an edge is erased it is removed from the map data structure. The inclusion tree is updated before each edge removal, using the same tests as above. When the type of the edge has been found, the inclusion tree is updated by performing the following actions:

```
switch (edge type)
{
  case 2: one inner boundary is split to give one inner boundary and one outer boundary
  case 3: two inner boundaries are merged to give one single inner boundary
  case 4: one outer boundary and one inner boundary are merged to give a single outer boundary
  case 5: delete an outer boundary
}
```

```

    case 7: one outer boundary is split to give two outer boundaries
}

```

the addition or the removal of an edge of type 1 or 6 does not modify the inclusion tree.

The method presented above allows incremental construction of planar maps. Consistency between geometry and topology are achieved through exact intersection of lines. Let us notice that this method is used in the context of a 2D drawing.

4.5. DATA STRUCTURES AND ALGORITHMS

As already mentioned, the data acquisition and the construction of the planar maps are made in three stages: data acquisition (input of the drawing), pre-processing of the drawing and construction of the planar maps.

4.5.1. INPUT OF THE DRAWING

We use the mouse to make the input of the drawing. The mouse sends, with a constant frequency, the coordinates of the points of the drawing. When the mouse button is up or down, a special event is sent. So the result of this stage is a list of points marked by the state of the buttons (UP or DOWN). This mode of acquisition has the advantage to reconstitute the direction and the order of the different parts of the drawing.

4.5.2. PRE-PROCESSING

Malformations coming from a freehand input may be subdivided into three categories. The first one is made up of input mistakes and big erasures that happen when the user didn't pull it off. These malformations are eliminated by hand, during the drawing. The second one puts together mistakes due to hand trembling of the user during the acquisition of the crude drawing. These mistakes will be called burrs. Their processing can be done in two stages: first a reduction of keyboarded points by a sampling process, and then an adjusting process. The third category is made up with the breaking of lines or the superposition of several lines. These malformations, which will be called erasures, cannot be corrected without context knowledge, as mentioned by (14). This last category will be taken into account during the planar map construction.

Let us notice that our software will be concerned only with straight lines. The treatment of curves (such as B-splines for example) is a lot intricate and some of the methods presented here are not suited for curves (for example, the appearance of curves can be completely modified by our algorithms).

4.5.2.1. SAMPLING AND ADJUSTING

A graphic table or a mouse sends a great number of points for each drawing, whereas a restricted number is enough to express appearance of lines. Several methods have been developed to sample a crude drawing. Berthod and Jancenne (7) use a smoothing method which can, in some cases, dismiss some angular points. We chose the method from (5). With it, the sequence of points corresponding to a straight line can be reduced to its two ends A and B.

However, for a given straight line, points given by the user are rarely lined up. Some small deformations (peaks) are involuntary introduced by the user. Correction of these peaks has to be done, while keeping intentional changes in the directions from the user.

The algorithm used is that of Belaid et Masini (5). The achieved input is then sent to the following stage as vectors joining successively kept points.

In a freehand drawing, it is important to assert that edges are parallel or orthogonal. The adjusting process replaces each sampled edge by an edge which has the same middle than the former and the nearest direction among the authorized ones. The algorithm used is that of Michelucci.

4.5.2.2. DELETION OF ERASURES

The kinds of erasures the most commonly noticed are represented on figure 13. These erasures are due to the fact that it is quite impossible to freehand draw a figure with perfect connections. To correct such erasures without contextual knowledge may be dangerous. For this reason, in the framework of project mmi2, these corrections will be done during the construction of the planar map data structure. Unfortunately, there does not exist any rule allowing the correction of such erasures. Thus, we used heuristic methods.

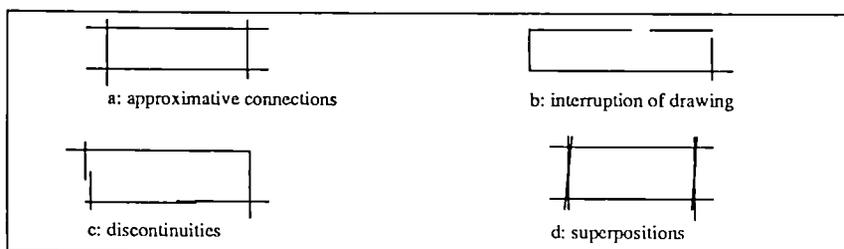


Figure 13: examples of erasures

The construction of planar map allows, for the first kind of erasure (see figure 13(a)), to delete ends of edges, systematically or by picking. Similarly, the planar map data structure allows to delete superposed edges or to merge the edges of discontinuity (see figure 13(c)).

Kind (b) of erasure can be eliminated by considering a circle with given radius: all the edge vertices lying inside such a circle will be linked up to a point inside the circle. This method may be applied to correct erasures of kind (b).

The three stages preprocessing proposed in our method (sampling, adjusting, removing of erasures) may be seen on figure 14. The first stage is applied to a crude drawing and allows to eliminate non significant points in order to make easier the management of the list of points and to simplify next stages. The adjusting stage allows to present to the user a more pleasant drawing. The third stage needs a contextual knowledge. In the framework of project MMI2, planar maps give a satisfactory solution for this problem.

4.5.3. THE CONSTRUCTION OF THE PLANAR MAP

An overall data structure of the planar map is defined in section 4.4. In the Network Diagram tool, the building and the network are represented by two different planar maps on which we have to attach attributes to support some specific information.

To each planar map, we attach a list of objects. For the building planar map, this list contains the technical shafts; for the network planar map, this list contains computers,

disks, taps and so on. When the user puts an icon representing an object on the drawing window, the instance of this object (a copy of the object) is attached to the list of objects of the network planar map.

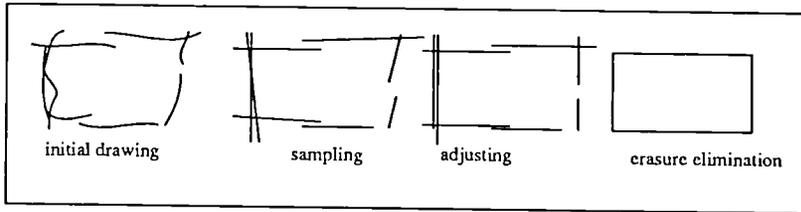


Figure 14: preprocessing of a drawing

Each object may be connected to another one; for example, a Sparc-SLC station may be connected to a disk. In fact, the connection between two objects is more complex. For example let us consider the connection between a computer and a tap (thinnet-tap-system): to establish such a connection, we need a transceiver and a twin cable. Thus the user must install the transceiver and the twin cable between the computer and the tap; this makes the task difficult and overloads the screen. Furthermore, Beatrice Cahour (1 and 2) indicated that when an expert draws a network on a sheet of paper, he represents the connections only by straight lines. Accordingly, in the network diagram tool the connections will be represented by segments, but the system has to add automatically the transceiver and the twin cable to the network list of objects. The transceiver and the twin cable are objects which have no graphic representation in the system.

The connection previously defined were established on the user's request. However, the Network Diagram tool must create links between the data structures representing the building and the network. For example, a link between a computer and the room containing it must exist in the data structure. Algorithms for such a search have been implemented.

REFERENCES

- (1) MMI2 Review, Literature and General Architecture, ESPRIT project 2474, BIM: Belgium.
- (2) MMI2 Review, ESPRIT Project 2474, Barcelona, 23d March 1990, BIM: Belgium.
- (3) D. AYALA, P. BRUNET, R. JUAN, I. NAVAZA, Object Representing by Means of Nonminimal Division Quadrees and Octrees, ACM Transaction on graphics, vol. 4, n? 1, pp 41-59, January 1985.
- (4) B.G. BAUMGART, A Polyhedron Representation for Computer Vision, In AFIPS conference, pp 589-596, 1975.
- (5) A. BELAID, G. MASINI, Segmentation de Trace Manuscrits sur Tablette Graphique en Vue de leur Reconnaissance, TSI vol. 1, n2, pp 155-168, 1982.
- (6) J.L. BENTLEY, T.A. OTTMANN, Algorithms for Reporting and Counting Geometric Intersections, IEEE, Transaction on Computer, vol. 28, n? 9, 1979.
- (7) M. BERTHOD, P. JANCENNE, Le Pre-traitement des Tracés Manuscrits sur Tablette Graphique, AFCET meeting, Reconnaissance des formes et intelligence artificielle, Toulouse, pp 195-209, September 1979.

- (8) J-L BINOT, P. FALZON, R. PEREZ, B. PEROCHE, N. SHEEHY, J. ROUAULT, M. WILSON (1990) Architecture of a Multimodal Dialogue Interface for Knowledge Based Systems, In Esprit '90 Conference Proceedings, pp 412-434, Kluwer Academic Publishers: Dordrecht.
- (9) R.CORI, Un Code pour les Graphes Planaires et ses Applications, Astérisque n° 27, 1975.
- (10) J.EDMONDS, A Combinatorial Representation for Polyhedral Surfaces, Notices of AMC, 7, 1960.
- (11) M.GANGNET, J.C. HERVE, T. PUDET, J.M.V. THONG, Incremental Computation of Planar Maps, Computer Graphics, vol. 23, n°3, July 1989.
- (12) Green, P. and Wei-Haas, L. (1985). The Rapid Development of User Interfaces: Experience with the Wizard of Oz Method. In Proceedings of the Human Factors Society - 29th Annual Meeting, 470-474, Human Factors Society: Santa Monica, CA.
- (13) Hobbs, J.R. (1985) Ontological Promiscuity. In Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics, Chicago, June 1985, 61-69.
- (14) C. HEROT, Graphical Input through Machine recognition of Sketches, Computer Graphics vol. 10, n° 2, pp 97-102, 1976.
- (15) C.M. HOFFMAN, J.E.HOPCROFT, M.S.KARASICK, Towards Implementing Robust Geometric Computations, In proceedings of the Fourth Annual ACM Symposium on Computational Geometry, ACM Press, New York 1988.
- (16) A. JACQUES, Constellation et Graphes Topologiques. In Combinatorial Theory and Applications, pp 251-260, Budapest 1970.
- (17) P. LIENHARDT, Extension of the Notion of Map and Subdivision of a Three-dimensional Space, In STACS'88, Lecture Notes in Computer Science 294, 1988.
- (18) J.M. MOREAU, Hierarchisation et Facetisation de la Representation par Segments d'un Graphe Planaire dans le Cadre d'une Arithmetique Mixte, These, Ecole Nationale Supérieur des Mines de Saint-Etienne, Saint-Etienne 1990.
- (19) D.E.MULLER, P.F. PREPARATA, Finding the Intersection of Two Convex Polyhedra, Theoretical Computer Science, vol. 7, n° 2, pp 217-236, October 1978.
- (20) M.H. OVERMARS, J.V. LEUWIEN, Maintenance of Configurations in the Plane, Journal of Computer and System Science, vol. 3, pp 166-204, 1981.
- (21) J.A. PITTMAN, Recognising Handwritten Text, in Human Factors in Computer Systems, Proceedings of ACM CHI '91, pp 271-276, April 1991.
- (22) M.SEGAL, Using Tolerances to Guarantee Valid Polygon reconstruction algorithm, Computer Graphics, vol. 24, n°4, pp 17-27, august 1990.
- (23) J. W. SULLIVAN, S.H. TYLER, (1991) Intelligent User Interfaces, ACM Press, Addison-Wesley: Reading, Mass.
- (24) Taylor, M.M., Neel, F. and Bouwhuis, D.G. (1989). The Structure of Multimodal Dialogue, North-Holland: Amsterdam.
- (25) W.T. TUTTE, Graph Theory, Addison Wesley, 1984.
- (26) A. VERROUST, Etude de Problemes Lies a la Definition, la Visualisation et l'Animation d'Objets Complexes en Informatique Graphique, these d'Etat, Universite de Paris-sud, 1990.
- (27) M.D. WILSON, A. CONWAY, Enhanced Interaction Styles for User Interfaces, IEEE Computer Graphics and Applications, vol 11, pp 79-90, March 1991.

THE TRANSFER OF VISION RESEARCH TO VEHICLE SYSTEMS AND DEMONSTRATIONS

B F Buxton*, S Arrighetti, D A Castelow, F Ferrari,
C G Harris, M Magrassi, S Masciangelo, P McLauchlan,
P Moron, S von der Nuell, M Rygol, G Sandini,
R Vaillant and H Wang

*GEC-Marconi Limited,
Hirst Research Centre, East Lane
Wembley Middlesex

SUMMARY

In this paper we focus on the development of the four experimental platforms and vision/vehicle demonstrations in the *VOILA* project. The main vision research activities in the project are outlined and areas of research of particular importance that enabled the platforms to be established are highlighted. The transfer of this research to the platforms is then described and the performance of the systems implemented discussed. In particular, it is shown how the successful transfer of some of the latest vision research to these experimental platforms and demonstrations and their implementation on multiprocessor computer systems has enabled a number of systems to be developed that provide concrete evidence that passive machine vision can provide the capabilities and performance required for this type of industrial application.

1. Background to the Project and Objectives

The *VOILA* Vision Research Pilot Project involves four companies: GEC, ELSAG, MS2i and Roke Manor Research; the French research institution, INRIA; and four leading university departments: the University of Genoa, DIST and DIF, the University of Oxford, Department of Engineering Science, and the AI Vision Research Unit at the University of Sheffield. Its aim is to develop dynamic vision systems for the control of robot vehicles operating either indoors, in industrial and commercial environments such as factories and warehouses, or outdoors, in stockyards, car parks and other less constrained environments.

In particular, in order to meet the industrial requirements laid down in the first year of the project a number of basic visual competences for obstacle detection and avoidance, free-space determination, local-navigation and global map-making have been developed. As described in the first annual report (1), the modules were then implemented on experimental platforms at four sites, at: GEC, MS2i, Genoa DIST and the University of Oxford. Each platform incorporating a robot vehicle, a vision system, and a special purpose multiprocessor computer system was established by the second project milestone in October 1990 and used to demonstrate the visual competences mentioned above. In addition, two of the platforms were adapted to work in "table-top" form for the ESPRIT Conference and Exhibition in November 1990. The remaining two will be used in a similar way to mount exhibits for this year's Conference.

Since milestone M2, experimental work has been underway to determine the performance of these systems and the Consortium has begun to integrate the software

developed on a central demonstration at MS2i in Montigny. This central, "integration" demonstration is being mounted on a *CAPITAN* multiprocessor architecture utilizing the infrastructure developed for the P1560 *SKIDS* project. In addition, the low-level vision system at MS2i is based on the *DMA* (Depth and Motion Analysis) system developed in P940.

2. Vision research and development of the experimental platforms

In the introduction above, we have briefly described the experimental platforms and demonstrations being developed within the *VOILA* project. However, this only represents approximately 40% of the Consortium's activity. Almost 60% of the project is devoted to vision research activities and the Consortium has produced over 80 papers for publication in the primary literature. The aim of this report is to briefly describe some of this research and to show how it has been successfully transferred to the experimental platforms and used to develop the demonstrations. In the following, we thus briefly outline the organization and main areas of research in the *VOILA* project, discuss a number of research activities in greater detail, and describe how these were transferred to the experimental platforms and the results obtained. In the final section we briefly comment on the industrial and commercial significance of our work.

2.1. The main areas of vision research in VOILA

In embarking on this research, the Consortium was able to build on previous work in preceding ESPRIT projects, especially P940, and other national projects. In particular, four major vision systems were available to the Consortium:

- *DROID* - a system based on point or corner features that progressively builds up and refines a 3D representation of what has been seen by utilizing relative motion between the camera and the scene.
- *CARVE* - a system developed in P419, Image and Movement Understanding, that is based on a pixel representation of the image and a volumetric representation of the scene. *CARVE* builds up a voxel description of what has been seen using a moving binocular stereo camera system.
- *TINA* - a binocular system that uses stereo fusion of edge features to build up a 3D representation of the world. These edge features can be further described in terms of line segments or circular arcs which can themselves be fused between stereo views or matched to CAD-like wire-frame models stored in a database for object recognition and location.
- *DMA* - a system based on the representation of edge features as line segments that can then be tracked from frame to frame by a token tracker or fused in a trinocular stereo system to build up a 3D description of what has been seen. This system is currently being implemented in pipelined hardware in P940.

Each of the systems has been further developed and enhanced within the *VOILA* project and all are being used in the development of the final demonstrations. In the next section we thus highlight areas of the vision research that have been particularly important in developing the demonstrations and describe how they have been transferred to the four experimental platforms.

3. Transfer of the research to the demonstrations

For the purpose of establishing the four experimental platforms and developing the final demonstrations the Consortium is divided into four subgroups as shown in table 1.

Subgroup A	Subgroup B	Subgroup C	Subgroup D
*Oxford MS2i RMR	*DIST ELSAG DIF	*GEC Sheffield	*MS2i INRIA ELSAG

Table 1: The four project subgroups set up to establish the four experimental platforms and October 1990, milestone M2, demonstrations. The "*" indicates the site of each platform whilst the "integration" partner, MS2i is underlined.

3.1. The Subgroup A, Oxford experimental platform

This platform is based on the *DROID* system, a GEC Electrical Projects research AGV at the University of Oxford and a special purpose multitransputer architecture. The *DROID* system works by extracting features (corner points) from digitised images and individually tracking these features between successive image frames using a Kalman filter (2). It can then compute the positions of feature points in 3D-space and subsequently refine these and estimates of the camera motion (ego-motion) using new observations. Over time, the trajectory of each corner feature in 3D-space is maintained and updated by *DROID*. The system thus consists of the following stages:

- corner detection
- refinement of existing 3D corner position estimates by matching to new 2D feature points
- estimation of ego-motion
- instantiation of new 3D points in space by using the ego-motion estimates to match previously unmatched points
- 3D scene reconstruction and interpolation.

Three main enhancements were made to *DROID* in order to implement it in parallel on the Oxford experimental platform:

- (i) Performing the corner detection in parallel on a transputer array;
- (ii) Arranging the timings between frame capture, corner detection and the remaining *DROID* processing;
- (iii) Providing real-time software control of the functioning and display modes of *DROID*.

The parallel implementation of *DROID* For the purposes of this parallel implementation an architecture known as *PARADOX* (3) was developed by splitting the *DROID* system into two parts: corner detection and the rest. In fact, corner detection is a data independent operation which accounts for more than 90% of the entire computation on a serial machine. The remainder of the system is data dependent with relatively higher-level data structures. In the *PARADOX* architecture, a Datacube MaxScan board is used to digitise 256 x 256 images from a camera on the robot vehicle. These are

simultaneously stored in buffers in two Datacube ROIstores. One ROIstore feeds a Digimax for display of a previously stored frame with a Sun workstation generating overlays that represent the inferred 3D geometric data output from *DROID*. The other ROIstore repetitively broadcasts each frame in a form suitable for reception by an interface board designed by the British Aerospace Sowerby Research Centre on the P1560, *SKIDS* project. This board has 512 kbyte of dual-ported video memory which can be accessed via the Maxbus and also by an onboard T800 transputer. The four serial links on this transputer thus enable a 256 x 256 stream of 8-bit image data to be passed at video rate to an array of 32 transputers on a Transtech MCP 1000 card.

The *PARADOX* architecture and the corner detection algorithm for execution in parallel on the transputer network were developed in the *VOILA* project. In order to implement the corner detection in parallel, a processor-farm approach was taken using a multiple one-dimensional array topology. The network of transputers, including the BAe T800, is divided into three functional parts: a server node, an image grabbing node, and computing nodes.

The server node acts as a communication interface between the Sun and the transputer network. It passes commands from the host to the image grabbing node on the BAe interface board. The extracted corner features are also passed back through the server node into the Sun. The image grabbing node receives commands from the host via the server node. Images are partitioned into 4 x 7 segments and are transmitted in parallel to the network where each computing node intercepts a segment and computes the corner locations. A Sun4 workstation acts as overall system controller and also carries out the 3D computations required in the *DROID* system.

Performance The performance obtained from the *PARADOX* architecture for parallel *DROID* was 0.87 seconds per frame which is 17 times faster than implementation on a Sun4 alone. The overall performance is limited primarily by the parallel execution of the 3D computations and the corner detection algorithm which have comparable execution times. The Datacube control and visual display contribute a negligible portion of time. This performance could be improved by off-loading some of the initial corner detection operations such as global convolutions onto the Datacube modules. However, the *DROID* algorithm requires 16-bit fixed-point precision fairly early in the corner detection algorithm which limits the possibilities for this. Another approach would be to parallelise the 3D computations and transfer some parts of it to a transputer network. Both of these are being considered at present.

The performance measures that can be made on *DROID* itself fall into two basic classes: ego-motion and structural representation. The accuracy of the ego-motion is simple to specify, but to measure it requires accurate ego-motion ground truth, which is often difficult to obtain. This question is currently being addressed using the Oxford vehicle which is equipped with a GEC Electrical Projects laser scanning guidance system capable of accurately determining the vehicle ego-motion (4). This has been used to show that estimates of the ego-motion from *DROID* suffer from long-term "drift", especially when sharp turns are executed by the vehicle. For example, although on the Oxford platform *DROID* appears to estimate the speed of the vehicle well when it is driven in a straight line (the speed in the first frame was supplied in order to overcome the depth-speed scaling ambiguity), its position drifts as shown in figure 1.

However, a stereo version of *DROID* also developed in *VOILA* is less susceptible to drift. For example, in one experiment a stereo rig was driven along a circular path of circumference 500m, and images captured every 0.4m. Performing stereo *DROID* processing resulted in the near circular path labelled "stereo" shown in figure 2, and an

absolute accuracy after completing one circuit of 3m. Using monocular *DROID* resulted in a significant scale drift (the path labelled "unconstrained mono" in figure 2), but when odometry in the form of the vehicle speed was utilised, the circular path was regained (the path labelled "constrained mono" in figure 2).

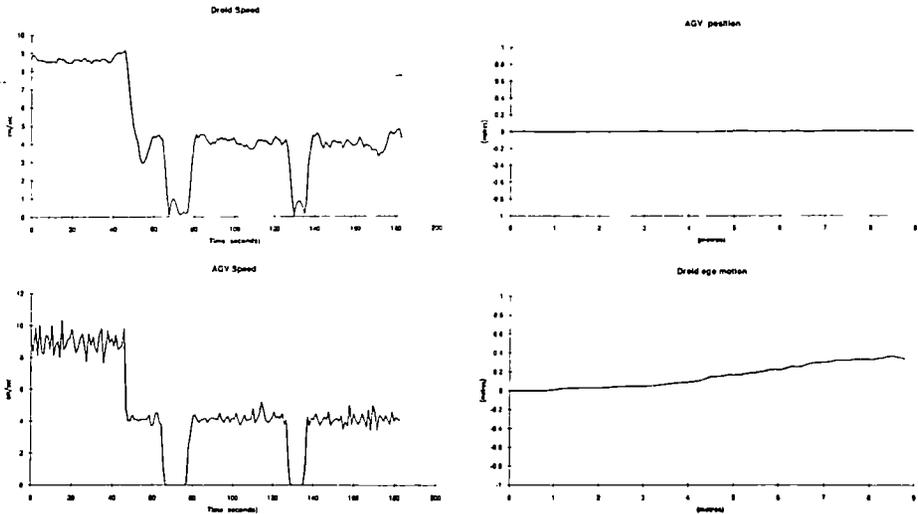


Figure 1: Estimates of the vehicle speed and position from *DROID* and from its laser scanning guidance system.

3.2. The Subgroup B. Genoa DIST Experimental Platform

This platform is based on a TRC labmate robot vehicle equipped with three cameras, two of which are used for a *reactive* real-time obstacle detection system (5) whilst the third is used to provide input to a monocular scene interpretation system that uses vanishing points (6) automatically to reorient the vehicle in an indoor environment such as a corridor. For the October 1990 demonstration, these two modules were implemented on modest hardware: a VDS 7001 *EIDOBRAIN* workstation equipped with special hardware dedicated to image acquisition and processing and a PC 486 equipped with an Imaging Technology acquisition board. The overall system is implemented as a subsumption architecture (7) in which the obstacle detector provides the lowest level of reactive behaviour for safe operation of the vehicle whilst the vanishing point system provides a higher level for maintaining the orientation of the vehicle.

The obstacle detection system is based on a fast comparison between the current stereo disparity and a reference map of the vehicle's ground plane so that everything higher or lower than the ground plane is treated as an obstacle. This technique was initially developed in P419 for a static camera system, but it was developed for the vehicle system within the *VOILA* project. In particular the obstacle *avoidance* strategy and the overall system architecture were developed within *VOILA* (8). Similarly, vanishing points had been studied by the Genoa DIF team in P940, but the demonstration module implemented was developed in *VOILA*. Here we shall concentrate on the obstacle detection and avoidance module and the system architecture.

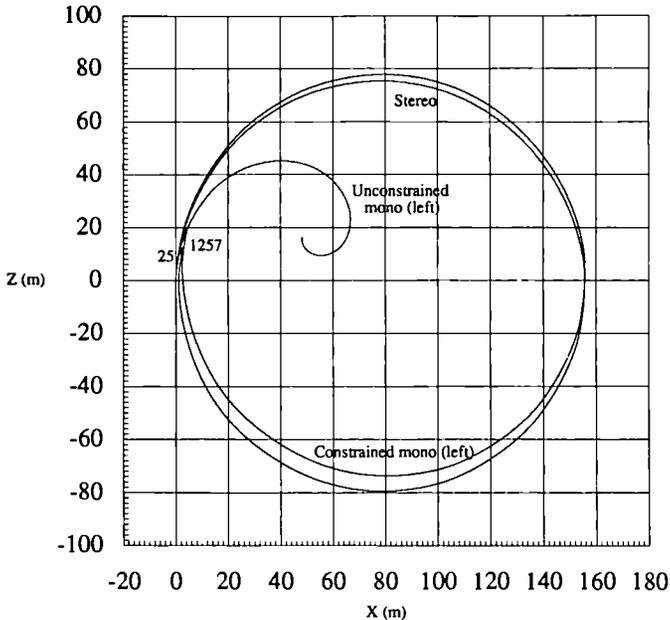


Figure 2: A plan view of the circular path of the vehicle in the outdoor experiment on *DROID*.

The obstacle detection and avoidance module The obstacle detector and avoidance module has first to produce a reference map of the disparity of the ground plane. A stereo algorithm based on a coarse-to-fine correlation procedure is used to compute this with the vehicle in front of a known clear area in which the floor has been artificially textured with magazine cuttings etc. At the coarsest level correlation is performed on 81, partially overlapped, square patches 32 pixels wide. After two successive refinements, a disparity measure is obtained on 8 pixel wide patches. This procedure is then repeated iteratively, changing the image of the pattern on the floor by moving the vehicle a little between views. A disparity map of a *safe*, flat piece of floor is thus obtained *without* explicitly having to calibrate the intrinsic or extrinsic parameters of the cameras.

Once this disparity reference map has been computed, the on-line algorithm is applied over three windows of interest in the stereo images:

- A central window is examined during motion to detect obstacles. Its minimum size is constrained by the physical dimensions of the vehicle.
- Two lateral windows are used to choose the best motor strategy for avoiding an obstacle detected in the central window.

In the on-line algorithm, a patch in the central window of the left image (say) is correlated with the corresponding patch in the right image shifted by the expected disparity of the ground plane. An obstacle is detected by comparing the value of this cross-correlation with a predefined threshold. A rough estimate of the obstacle's position is also obtained so that the system can decide which of the lateral windows should first be checked for an alternative route. Since there is no reference to the shape or form of the obstacle, this procedure is generic over obstacles and works for static or (slowly) moving objects.

Implementation of a subsumption architecture on a multiprocessor machine

In preparation for a final "teleguidance" demonstration, this ground plane obstacle detector has also been implemented on an *EMMA2* multiprocessor architecture at ELSAG. The *EMMA2* configuration used consists of two acquisition boards, a three-processor (286/ALA2) board (PN), and a family supervisor processor (P1). The three-processor module (PN) is used in the computational part of the algorithm. Each of the 3 Intel iAPX286 is used to perform the same task by means of a data partitioning approach. In addition there is a level of temporal parallelism in the implementation: a pipeline method allows the P1 processor to acquire new stereo images while the PN module is still processing the previous ones. The synchronization between P1 and PN is based on a semaphore mechanism that is a service offered by the *EMMA2* operating system.

Furthermore, the *EMMA2* machine is well suited to implementing the subsumption architecture developed for the *telerobotic* control of this platform in the final demonstration (9). *EMMA2* is an *MIMD* machine based on a common-bus parallel processing system, consisting of a multilevel, hierarchical structure (10). At the top level, the *EMMA2* architecture consists of a network of multiprocessing machines called *regions*, connected by high-speed, point-to-point data links and communicating with each other by message exchange. Each region contains a two level shared memory bus structure, in which the top level is the region level, which interconnects low level multiprocessor machines called *families*. The machine structure is based on memory hierarchy in order to minimize bus occupation.

The basic idea in implementing the subsumption architecture is thus to employ different families and allocate a single well-defined competence to each family. In this way each layer of the control architecture corresponds to a family of the hardware architecture. In addition to these families, there is a single processor used to control, through a serial line, the mobile platform and another processor which is in charge of implementing the subsumption. It simply simulates a kind of multiplexer which polls the various competence levels. During the normal navigation of the vehicle the subsumption processor enables the highest level to send control commands to the pilot, and inhibit all the other levels. When an emergency situation arises, for example from the lowest level, the *subsumption processor* disables all the other levels and enables the lowest level which becomes directly connected to the pilot.

Performance Particular attention is being devoted to the testing of the performance of the GPOD under different illumination conditions and for different obstacles. The parameters involved in the testing are:

- minimum height of a detectable obstacle;
- robustness to variations in illumination conditions;
- time necessary to "react" to obstacles;
- the minimum ground slope that would be detected as an obstacle when calibration has been carried out on a flat surface;
- the influence of the texture of an obstacle on the module's performance.

Experiments are still in progress, but the main results can be summarized as follows:

- The minimum height detected on a grey background, using a circular black pattern with a radius of 4 centimeters, is 7 cm.

- The time elapsed from the detection of an obstacle and the issue of the relevant motor command is 160 msec (which includes the choice of the turn direction based upon the evaluation of two lateral windows).
- The angle measured for the minimum slope detected as an obstacle is 5.44 degrees.

In addition, the problem of the on-line equalisation of the stereo pair has been addressed and solved. This enables differences between the two input images caused by different sensor responses or non-uniform illumination to be eliminated.

3.3. The subgroup C, GEC experimental platform

This platform is based on the *TINA* vision system developed at the University of Sheffield (11), a second (but later) version of the GEC Electrical Projects research AGV at the GEC-Marconi, Hirst Research Centre, and the *MARVIN* multitransputer architecture developed in a separate collaborative SERC/ACME project by GEC and Sheffield (12). This machine contains several special purpose *TMAX* cards each containing 1 MByte of dual ported videoRAM and a T800 transputer that provide a wide bandwidth, multitransputer interface between data on the commercial Datacube, MAXBUS, video-bus and a transputer array. *MARVIN* is used to carry out the computationally demanding image processing and geometric calculations required in the *TINA* system. However, although full-frame stereo, including the low-level Canny edge detection, stereo fusion, line fitting and object recognition and location can be performed in approximately 10 seconds on the 24 transputer *MARVIN* at Sheffield or in 12-15 seconds on the 18 transputer *MARVIN* at the Hirst Research Centre, this is still at least an order of magnitude too slow for a machine to maintain an on-going description of its environment. For example, since the AGV moves at speeds of 1 m/sec or more, it is necessary to recover scene geometry and locate objects at rates exceeding 1 Hz if objects are to be tracked and their position and orientation continuously maintained as the vehicle moves.

Predictive feed-forward stereo tracking Thus, an alternative approach was adopted that could exploit the spatio-temporal coherence of the world. A prototype research system was first developed using the Sheffield *MARVIN* machine and their local research vehicle known affectionately as *COMODE*. In this prototype, the full-frame stereo system described above is used to recognise and locate objects and to "boot" a run-mode in which the *predicted* position of an object is used to reduce the amount of processing required (13). The resulting predictive feed-forward stereo-tracking algorithm is able to return the 3D position and orientation of a moving object every 200 ms on the Sheffield system and approximately every 300 ms on the smaller HRC *MARVIN* machine.

The algorithm is implemented on the *MARVIN* architecture so that edge features represented as 3D line segments are tracked in parallel and the segments successfully tracked are used to estimate the position and orientation of the object as a whole. The tracker is structured internally as a *processor farm*. The master of the farm is known as the virtual tracker and each worker is known as a feature tracker. Each feature tracker is resident on one of the *TMAX* cards in order to ensure the most rapid access to the video stream of stereo image data. To speed up the processing, each feature tracker performs a 1D edge detection process in the vicinity of the predicted position of the line segment to find the new actual position of the object's edge feature. New line segment descriptions of the edge are then computed for both stereo images and projected out

into 3D as usual. In order to implement these processes efficiently, the virtual tracker itself contains three parallel processes. First, a *prefetcher* which obtains vehicle odometry (if required), grabs the next pair of images, and enables the transmission of new images to the *TMAX* cards. Second, a *farmer* that sends features into the processor farm, and lastly, a *reaper* process that collects results from the farm and instructs the farmer to send out new features for tracking. When all features have been returned, tracked or otherwise, the position and orientation of the object are computed and the latest position of a moving object may be added to the display as shown in figure 3.

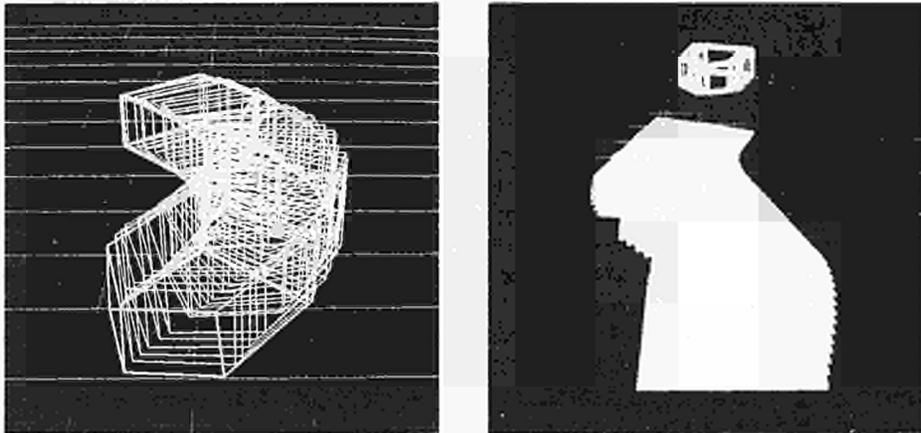


Figure 3: An example of tracking a known object on the Sheffield prototype. The object is moving towards the vehicle and the display updated once a second.

The above system was developed in prototype at Sheffield (14), ported to the HRC experimental platform (15) and was first demonstrated in October 1990. More recent experiments at Sheffield have enabled the *COMODE* to dock with a known object (16) and to successfully follow a moving object or "carrot" through a door.

Free-space determination In addition to these object recognition, location and tracking competences in which the system knows about the object or objects being located and tracked etc., a free-space determination module was also developed to support the navigation of the vehicle. It uses the 3D line segments obtained from the full-frame stereo process. In this case, however, the line segments are projected into the ground plane of the vehicle, both vertically and from the camera viewpoints, so as to obtain the *obstructed* and *obscured* regions into which it may not be safe to drive the vehicle. Since the field of view of the cameras is usually limited to 20-25 degrees, a map of the obstacles in front of the vehicle is built up by mounting the cameras on a turntable and taking several views. Detection of obstacles down to as little as 20-25mm in height (below which it is deemed safe for the vehicle to run over them) can be made robust by insisting that obstacles should appear consistently in at least two views (16). Furthermore, as noted in section 3.2, the obstacle detection is *generic* over objects: all that the system needs to know is the position of the ground plane which is obtained by placing a calibration tile on the floor prior to the free-space determination.

Again, this system has been successfully implemented on the GEC platform and was first demonstrated in October 1990. Since then, it has been extended so that views from different positions of the vehicle can be compared. Using this enhanced system, recent

experiments have shown that the laser scanner on the AGV guidance system had a systematic error of 3 degrees in the orientation of the vehicle. Furthermore, when the vehicle was driven around and then returned to its original position as determined by marks on the floor etc., it was found by superimposing the views of the edge segments obtained from the two vehicle positions (figure 4) that the position reported by the laser scanner could, as expected from the vehicle's on board Kalmar filters, be in error by up to 20mm. The vision system could detect this error to an accuracy of 2mm, about an order of magnitude better than the accuracy laid down in the industrial requirements (1,17).

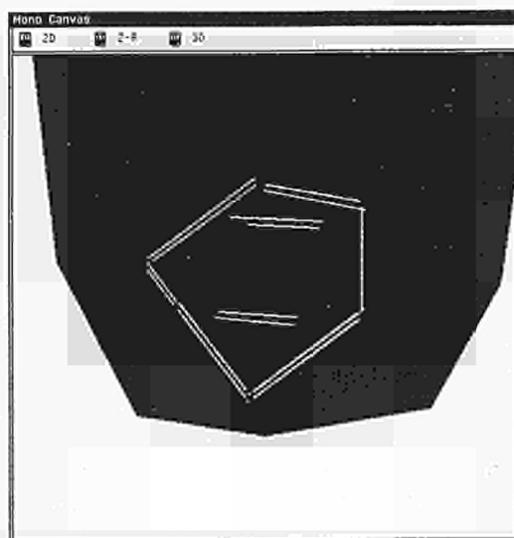


Figure 4: Two views of the edge features of an object seen from *nominally* the same vehicle position.

3.4. The subgroup D. MS2i experimental platform

This platform, which also serves as the baseline for the central software integration, was described in the first annual report (1). The machinery and hardware used have already been mentioned in the introduction in section 1. Of particular interest here however is the fact that in establishing this platform a deliberate attempt was made to develop a system for object acquisition and modelling that could deal with smooth, curved shapes and so extend the range of objects that stereo vision systems could deal with. Thus, the system is primarily based on research on detecting the extremal boundaries of curved objects and estimating their surface curvature. This information is supplemented by the detection of elliptical arcs in images and their perspective inversion into 3D circular arcs. Together, these modules have been used to develop a robust cylinder detector and modeller. However, below we focus attention on the former, extremal boundary work, which is based on the trinocular stereo vision system implemented on the *DMA* machine.

Detecting extremal boundaries and modelling curved surfaces The principle underlying the detection of an extremal boundary is simply that the position of an occluding contour or extremal boundary on an object depends on the position of the

camera (18). This is because, unlike the edge features fixed on the creases in the surface of a polyhedral or piecewise smooth object, they are defined by the condition that the line of sight lies in the tangent plane to the surface. Each pair of cameras in a trinocular vision system would thus predict that edge segments on an extremal boundary lie in slightly different positions in space - the difference being larger the larger the radii of curvature of the surface. Conversely, this difference can be used to estimate the curvature of the surface. This is first done under the assumption that the surface is locally cylindrical (19) and the radius of curvature computed. If the calculated curvature is very small the edge is considered to be an ordinary crease boundary but if it is large, it is concluded that an extremal boundary has been detected and, in addition, that a component of the surface curvature has been estimated. What is "small" and what is "large" is determined by a threshold that depends on the accuracy of the image line segments - a parameter that can be determined in the fitting procedure used to obtain the line segments from the image data. In fact, although only the algorithm for cylindrical surfaces is used in the system developed for the experimental platform, the analysis can be extended to enable the second differential form of a free-form curved surface to be estimated (19).

Implementing the cylinder detector and modeller On the experimental platform, a cylinder detector and modelling system was developed by combining this research on extremal boundaries with work on the perspective inversion of elliptical arcs detected in the images (20). By combining these two modules, each of which gives an *independent* estimate of the axis of a cylindrical object, and by combining information from several views of an object as the camera is moved, a robust cylindrical object detection and modelling system was built. For example, as shown in table2 almost all the cylindrical objects used in an experiment, even the most difficult highly reflective ones, could be correctly identified when results from four camera positions were combined.

Views	(a)	(b)	(c)
	%	%	%
1	99	66	40
2	100	90	70
3	100	99	86
4	100	100	98

Table 2: Percentage of cylindrical objects correctly identified for: (a) perfect geometric or synthetic objects; (b) only slightly imperfect objects such as cups, jugs, mugs etc. with few reflexions and no metal edges; (c) reflective objects such as cans and bottles made of plastic, metal or glass and bearing many surface markings such as writing etc.

These results were obtained under good illumination conditions on a controlled test rig. Under these conditions, with a calibration error of at most 0.25 pixel, the equivalent 3D precision is 1 pixel. However, on the MS2i vehicle system with a shorter focal length camera (8mm as opposed to 16mm), a larger stereo rig and a larger working distance (4m instead of 2m) and a maximum calibration error of 0.5 pixel, the equivalent 3D precision is 5-10 pixels.

4. Industrial and commercial significance of the results obtained

The final demonstrations in the *VOILA* project are set to target three areas for the application of vision systems to the control of robot vehicles; teleguidance of a robot vehicle; the operation of AGV systems in factories and warehouses; and the outdoor operation of robot vehicles in "natural" surroundings. In all three cases, the sensory capabilities of current systems are relatively primitive and the degree to which *flexibility and autonomy* may be combined is low. For example, in current teleoperation systems, the human operator usually provides a high degree of flexibility by having direct *low-level* servo control of the vehicle but there is no autonomous operation of the vehicle. Such systems are therefore frequently slow and very tiring to use and communication delays and failures can be very troublesome or even disastrous. Conversely, current AGV systems may be able to obtain a high degree of autonomy by utilizing guidance beacons such as buried wires, reflective strips fastened to the floor or bar-codes located at known positions, but they lack the flexibility to operate in an unconstrained or more natural environment.

In both cases, the primitive sensory capabilities of current systems is the underlying reason for their limited performance. The sensory capabilities they have fail to give the robot vehicles the kind of detailed perception of their environment they need, in particular, for safe operation in peopled environments or environments designed to be used by people (1). Whilst it is widely accepted that passive vision is one of the very few sensory systems capable of delivering the kind of detailed information required (sonar - as in bats may be another) the following doubts have been raised as to the applicability of vision.

- (a) Machine vision (currently) lacks the required technical capabilities (*it cannot do it!*).
- (b) Machine vision is too slow, too computationally intensive and too expensive.
- (c) Other sensors will do the job more cheaply.
- (d) Vision is insufficiently robust and not industrially proven.

These doubts, especially the last three, have been raised previously and to some extent answered in general terms (17). However, by establishing the four experimental platforms in the *VOILA* project, successfully transferring some of our latest research to them and assessing the results quantitatively, we have been able to provide specific answers to (a) and in part to (b), (c) and (d). In particular we have been able to demonstrate what state of the art machine stereovision systems can be capable of. Furthermore, by utilizing multiprocessor computer systems, we have shown that machine vision need not be slow, even in experimental systems. Since, as reported in (17) speed improvements of a factor of five can be expected when moving from such development systems to dedicated hardware and since there are technological advances already in train, such as development of the T9000 transputer and improved floating point DSP boards, we can be fairly certain that the requisite computational power can be provided much more cheaply than hitherto thought possible. Finally, our experiments show that in some cases, vision can out-perform other sensors, and can even be very sensitive to small positional errors. In particular, when used for obstacle detection, vision can be both sensitive, fast and robust.

Acknowledgements

In writing this paper, the authors wish to thank the members of the *VOILA* Consortium for their help and support.

References

- (1) B. F. Buxton and M. T. Roberts, "P2502 VOILA Vision Research Pilot Project, First Annual Report, April 1989 - March 1990, The demonstration and industrial evaluation of vision/AGV systems," May 1990.
- (2) C.G. Harris and J. M. Pike, "3D positional integration from image sequences," *Image and Vision Computing*, 6(2), 87-90, 1988.
- (3) H. Wang and C. Bowman, "The Oxford parallel architecture for 3D vision," *IEE Parallel Architectures for Image Processing Applications*, April 1991.
- (4) P Stephens, US patent 4,647,784, 1987.
- (5) F. Ferrari, E. Grosso, G. Sandini and M. Magrassi, "A stereo vision system for real time obstacle avoidance in unknown environment," *Proc. IEEE Int. Workshop in Intelligent Robots and Systems; Tsuchiura, Ibarachi, Japan, July 3-6, 1990*.
- (6) P. Bellutta, G. Collini, A. Verri and V. Torre, "3D Visual Information from vanishing points," *IEEE Workshop on Interpretation of 3D Scenes, Austin TX, USA, November 27-29, 1989*.
- (7) R. A. Brooks, "A Robust Layered Control System for a Mobile Robot," *IEEE Journal of Robotics & Automation* 2:14, 1986.
- (8) M. Fossa, E. Fuiano and G. Sandini, "A vision architecture for navigation," *Proc. of Int. Conf. Intelligent Autonomous Systems 2, Amsterdam, 11-14 December 1989*.
- (9) F. Ferrari, M. Fossa, E. Grosso, M. Magrassi and G. Sandini, "A Practical Implementation of a Multilevel Architecture for Vision-based Navigation," *Proc. 5th Int. Conference on Advanced Robotics, Pisa, 19-22 June, 1991*.
- (10) E. Appiani, B. Conterno, V. Luperini and L Roncarolo, "EMMA2, A high performance hierarchical multiprocessor," *IEEE Micro* 9(1), 42-56, 1989.
- (11) J. Porrill, S. B. Pollard, T. P. Pridmore, J. B. Bowen, J. E. W. Mayhew and J. P. Frisby, "TINA: a 3D vision system for pick and place," *Image and Vision Computing*, 6(2), 91-100, 1988.
- (12) C. R. Brown and M. Rygol, "*MARVIN*: Multiprocessor architecture for vision," *Proc. 10th. Occam User Group Technical Meeting, 1989*.
- (13) S. B. Pollard, J. Porrill and J. E. W. Mayhew, "Predictive feed forward stereo processing," *Proc. Fifth Alvey Vision Conference, Reading, U.K. 25-28th Sept 1989, pp97-102*.
- (14) S. B. Pollard, J. Porrill and J. E. W. Mayhew, "Experiments in vehicle control using predictive feed-forward stereo," *Image and Vision Computing*, 8(1), 63-70, 1990.
- (15) B. F. Buxton, D. A. Castelow, M. Rygol, P. McLauchlan, P. Courtney, and S. B. Pollard, "Developing a stereo vision system for control of an AGV," *IEE Second International Specialist Seminar on 'The Design and Application of Parallel Digital Processors,' The Gulbenkian Foundation, Lisbon, Portugal, 15-19 April 1991*.
- (16) M. Rygol, P. McLauchlan, P. Courtney, S. B. Pollard, J. Porrill, C. R. Brown and J. E. W. Mayhew, "Parallel 3D vision for vehicle navigation and control," *Transputing'91, World Conference, Sunnyvale, California, 22-26 April 1991*.
- (17) M. T. Roberts and B. F. Buxton, "P2502 VOILA Vision Research Pilot Project, Evaluation and assement of vision/AGV systems and demonstrations for the VOILA Consortium," presented at the ESPRIT Computer Vision Workshop at ECCV'90, Antibes, April 23, on the Assessment of Computer Vision Systems," April 1990.

- (18) R. Vaillant and O. D. Faugeras, "Using occluding contours for 3D object modeling: Cylindrical objects," First International Advanced Robotics Programme Workshop, Toulouse, France, Oct. 1989.
- (19) R. Vaillant, "Using occluding contours for 3D object modelling," Computer Vision - ECCV90, Proceedings of the First European Conference on Computer Vision, Antibes, France, 23-27 April, 1990, pp 454-464, Springer-Verlag, Lecture Notes in Computer Science, 1990.
- (20) S. Masciangelo, "3D cues from a single view: detection of elliptical arcs and model-based perspective backprojection," BMVC90, Proceedings of the British Machine Vision Conference, Oxford, September 24-27, 1990, pp 223-228.

INTERCONNECTION NETWORKS FOR UNIVERSAL MESSAGE-PASSING SYSTEMS

Axel Klein
Siemens AG, ZFE IS SYS 31
Otto-Hahn-Ring 6
W-8000 Munich 83
Germany
Tel + 49 (0)89 636-41006

SUMMARY

The paper presents performance evaluation results and ensuing recommendations for the design and architecture of future transputer-based networks for scalable, universal message-passing machines. The main method used for the investigations was the simulation of network and load models on the packet level, partly extended by a novel combination with analytical models. After introducing the design space consisting of three major topology classes in a variety of configurations, we describe three important steps of performance evaluation and their results. In the first step, sustained random communication performance is evaluated regarding the prediction of throughput and delay, their variation with the network size and a cost-relative comparison of the different topologies. In the second step, various routing schemes are investigated for their performance effects on different load patterns. Finally, the effects of hot-spot load and of network partitioning are presented and discussed. The paper concludes with a strong recommendation in favour of multistage networks and the newly proposed adaptive routing scheme.

1. INTRODUCTION

In order to scale parallel computer systems into the range of hundreds or thousands of nodes without running into a memory bandwidth bottleneck, distributed memory architectures are generally regarded to be the appropriate solution. In the same way, although other communication paradigms may be more desirable from a programmer's viewpoint, message-based communication is currently regarded to make the most efficient use of distributed memory machines.

While in special purpose machines message communication can be optimized by the application specific choice of the network topology, by an optimal mapping of communicating processes to adjacent processors and sometimes even by a pipelined, systolic communication scheme, the interconnect system of parallel machines for general purpose use can (and should) not rely on such regularity properties of its target applications, because they are either not present at all or differ widely between the various application types. Such general purpose or universal message-passing systems require instead a network which supports arbitrary communication patterns equally well with a minimum of contention, and which is able to scale its performance with the network size closely to the theoretical optimum: i.e. linear increase of (sustained) throughput and (only) logarithmical increase of message delay.

The appearance of commodity communication components, which support flexible

and fast message routing in networks of any desired shape and size and which can be employed in arbitrary numbers to achieve any required throughput level (restricted only by the communication bandwidth of the terminal nodes), greatly facilitates the task of constructing such networks. In the PUMA project, such a communication component is currently being developed by Inmos as a dynamic packet router for the communication links of the new T9000 generation of transputers, the C104 switch which dynamically connects any number of virtual channels on 32 physical links with a peak bandwidth of 10 MByte/s per link and a sub-microsecond switching delay (3).

Since there is a huge design space for networks built from these switches, detailed knowledge is required on the relation between architectural parameters and the target evaluation criteria such as cost, performance and scalability, in order to guide the design decisions for a universal message-passing network properly. Therefore, it was another major task in the PUMA project to investigate and show up these relations and derive corresponding recommendations for system designers. It is the purpose of this paper to give an overview of these investigations and present its main results.

Basing on the new C104 switch and its anticipated properties, such as dynamic packet switching, wormhole routing, interval labelling and universal routing (3), we classified, specified and investigated the spectrum of relevant network structures for universal message-passing systems, which comprised

- to define a range of regular and scalable topology classes and their main configuration variants with respect to cost and performance, and to specify the corresponding deadlock-free routing schemes to be implemented with the given mechanisms
- to identify the characteristic performance criteria and produce performance predictions for the considered networks and a wide range of network sizes and load cases
- to determine the scaling behaviour of performance and cost with the network size
- to comparatively evaluate the topology classes and configuration variants with respect to performance, scalability and cost/performance ratio

In the subsequent sections we describe the design space investigated and the investigation methods used, present the main results on performance, cost and scaling properties for the various networks individually and comparatively, and discuss the use of non-deterministic routing schemes to support universal scalability. As a result, we propose a novel, adaptive routing mechanism which has been included into the C104 specifications. Finally, we present particular performance effects of hot-spot load and network partitioning.

2. DESIGN SPACE FOR SCALABLE NETWORK STRUCTURES

In order to make sufficiently detailed performance evaluations of network topologies and configuration variants, we had to restrict the evaluation space in a reasonable way. This restriction was guided pragmatically by criteria of technical construction, which requires regularity, modularity and a sufficient range of size scalability, and of compatibility with the interval routing mechanism (4), which requires the existence of an optimal or near-optimal, deadlock-free labelling scheme. This did, for example, exclude the fully connected networks of switches which cannot be scaled beyond 272 ports, or the direct butterfly networks which cannot be labelled in a well-balanced way. We therefore restricted our investigations to three major topology classes: the indirect multistage networks of the Clos-type, and the direct network classes of binary n -cubes and fixed-dimension grids. For regularity reasons, the considered configuration variants

were restricted to those with uniform link width (when multiple links were used), and we did not permit unconnected links (except the border links of the grid structures) in order to provide a fair comparison base of performance and cost between the different networks.

Although transputers have four links each, it is sufficient to consider networks with only one link per terminal node, as is done in the following investigations. Since "good" networks, which are able to scale their throughput (almost) linearly with the number of terminal links, necessarily have to increase the number of switches more than proportionally, the full communication bandwidth per transputer can most efficiently be used by a replication of the entire network for each of the four transputer links rather than by a four-fold increase of the single-network size. Such a replication additionally introduces an easy way to circumvent link or switch failures and to separate priority communication from other traffic.

INDIRECT NETWORKS

As indicated above, we consider the class of Clos-type multistage networks (in the following briefly termed *Clos networks*), where each switch of the first and last stage is connected with each switch of the center stage. The main advantage of this construction is, that multiple paths exist between any pair of terminal nodes, which are determined by the number of switches in the center stage, and which can be fully exploited by the routing scheme without creating deadlocks. Since the size of the center stage switches has to grow with the number of first stage switches, the three-stage limit (for networks built from 32-way crossbars) is reached with 32 first stage switches. For larger systems the center stage switches must be constructed recursively as multistage sub-networks.

Due to the bidirectional nature of the link connections and the C104's capability to independently switch unidirectional virtual channels, the logical multistage network can be folded at the center stage and mapped to $(s + 1)/2$ layers of physical switches (where s is the number of logical stages). Fig. 1 shows a 3-stage Clos network built from two layers of C104 switches, where the input and output links of the terminal ports are connected as bidirectional links to the switches of the first layer which corresponds to the first and last stage of the logical 3-stage network.

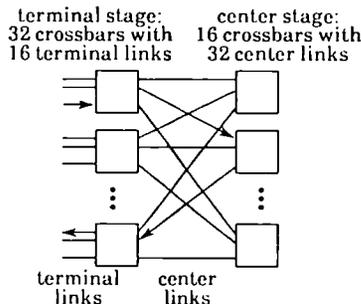


Fig. 1: 3-stage Clos network with routing example

The example is shown with an equal number of links on both, the terminal and the center side of the first layer switches. Configuration variants can now be generated by varying the ratio between the terminal and center links of the first layer switches (*link ratio a:b*). While a higher number of center links leads to a higher number of (redundant) center stage switches and hence, less contention and higher throughput can be

expected at the expense of increased network cost, a higher number of terminal links per switch reduces the number of remaining center links and represents a cheaper network with reduced performance. It should be noted that the more expensive networks with less terminal links per switch also reach their 3-stage limit of 32 first layer switches at a smaller network size or, in other words, must add further switch layers at smaller size increments.

By adding layers of switches, the Clos networks can, in principle, be scaled to arbitrary sizes (although the physical connection between the most distant switches of adjacent stages will set a technical limit to that expansion). Tab. 1 demonstrates the relation between network size (i.e. number of terminal ports), number of (logical) stages and number of switches. It must be noted, that for the more expensive networks the center stage sub-networks needed for the larger network sizes may be built with a 1:1 link ratio (instead of extending the expensive ratio of the first layer switches), thus achieving a more reasonable cost scaling. By varying the link ratio, the network throughput for a given size can be scaled to any required level up to the full crossbar performance. In the extreme case, however, this will require a huge number of switches which scales with the square of the network size like a full crossbar.

network size (# terminal ports)	link ratio (in all stages)						ratio first stage / next stages			
	16:16		8:24		4:28		8:24 / 16:16		4:28 / 16:16	
	#st	#sw	#st	#sw	#st	#sw	#st	#sw	#st	#sw
64	3	6	3	14	3	30				
128	3	12	3	28	3	60				
256	3	24	3	56	5	904			5	232
512	3	48	5	400	5	1808	5	208	5	464
1024	5	160	5	800	7	25312	5	416	5	928
2048	5	320					5	832	5	1856

Tab. 1: Numbers of stages (#st) and switches (#sw) for Clos networks of given size)

Interval labelling for the Clos-type networks is straightforward for the output links leading from the center stage towards the terminal outputs, provided that the terminal outputs are numbered consecutively. Since for the route from the terminal inputs towards the center stage all outputs can be arbitrarily chosen, an optimally balanced labelling scheme is achieved by equally dividing the complementary address interval over the center links. This kind of routing is deadlock-free by passing the logical stages in strictly increasing order and using separate links for each logical stage.

The random and adaptive routing mechanisms of the C104 can be efficiently exploited in Clos networks by routing arbitrarily from the first to the center stage and continue deterministically from there. In this way a dynamically even distribution of packets over all center switches is achieved regardless of the destination pattern. (This is discussed in more detail in section 4 of this paper). Deadlocks cannot occur in these schemes, because the stages are still passed in ascending order. In the random case, however, packets are always routed through all stages and do not exploit possible shortcuts implemented by the deterministic or adaptive routing scheme.

DIRECT NETWORKS

Direct Networks are described by graphs where each node is composed of one

switch with one or more transputers connected to it by one link each (terminal links of the switch). The remaining links of the switch are connected between adjacent nodes to form the edges of the graph. Edges may consist of multiple links which need not be distinguished in the graph topology, but present only a performance parameter. In a regular network topology, all nodes provide the same number of edges (*node degree*) and all edges have the same link width. While the topology class is determined by the topology of edges between nodes, the configuration variants are distinguished by the number of terminal links per node (*node size*) and the number of links per edge (*link width*). In the same way as for the indirect networks, the configuration variants allow to adapt a given topology to specific performance or cost constraints. Taking into account our restriction to configurations without unconnected links (for comparison purposes), we obtain a unique relation between the two configuration parameters. Fig. 2 presents the general model of a network node and its parameters.

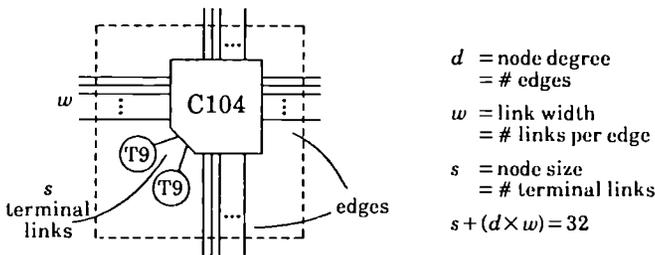


Fig. 2: Direct network node and its configuration parameters

The two most important classes of direct network topologies (as they have been used in many real system implementations) are the binary n -cubes or hypercubes and the k -ary n -cubes or grids. Although binary n -cubes can be regarded as a subclass of k -ary n -cubes, we view them as separate classes distinguished by the fact, that the latter (termed as *grids*) are scaled in size through variation of the width k for any fixed selection of the dimension n , while the former (termed as *cubes*) are scaled by their dimension.

Since the number of links per switch is bounded, so is the number of edges per node which sets an upper limit to the dimension of a cube. For small node size and link width this does not impose a severe restriction: cubes with up to 32000 nodes can be built with double links per dimension. It does, however restrict the possibility to scale the performance of a cube for given large size by extending its link width. Grids of low dimension, on the other hand, can be scaled to arbitrary sizes due to their fixed node degree and can have larger absolute link width, but are also performance-limited for large sizes since they would need to increase their link width with growing size in order to maintain their performance level.

The requirement for deadlock-free interval labelling leads to the restriction that grids must leave their border edges unconnected, i.e. the routing scheme can not make use of link cycles in each dimension, which effectively excludes the popular torus-grids from further consideration. Except for this restriction, the cubes and grids can be labelled to obtain an optimal and deadlock-free deterministic routing scheme. In order to make use of the C104's random routing mechanism to implement Valiant's two-phase routing scheme (8) (which attempts to achieve a balanced distribution of communication traffic across all links and thus to equalize packet delays) deadlocks must be excluded by using separate sets of links for each routing phase. In this way, two-phase random

routing (also termed *universal routing*) is restricted to network configurations with a link width of at least two. A fair comparison with other routing schemes would of course require the use of duplicated links in both cases. In section 4 we report simulation results for such a comparison.

3. BASIC EVALUATION OF NETWORK PERFORMANCE AND COST

In order to achieve our objective to determine the relation between the topology and configuration parameters and the performance, cost and scalability properties of the considered networks, we carried out an extensive performance analysis by means of simulation and analytical modelling. The simulations were performed with a proprietary, event-driven simulator tool (7), where the networks were modelled on the level of the packet transfer protocol and the corresponding features of the new transputer links and switch components. Hence, the dynamic effects of routing, buffering, flow control and internal or external contention are taken into account with sufficient accuracy to produce realistic performance predictions. Since runtime constraints of the simulator did not permit reasonable simulations with much more than thousand ports, analytical models which were fitted to the simulation results for smaller networks served to extend these results beyond the simulator limits.

Our investigations aimed at the suitability of networks for universal message-passing, where "universal" denotes the ability to cope with arbitrary communication patterns which are not particularly matched with the network topology. This ability can most effectively be tested (and compared) with sustained, non-systematic load patterns (e.g. random load), or with systematic but extremely unbalanced patterns (e.g. permutations or hot-spot patterns). Networks (or routing schemes) which perform well for these extreme cases, can reasonably be expected to do so for most realistic applications without requiring to find an optimal mapping between communicating processes and physical processors.

In the following subsections we present and explain performance predictions and comparisons for the considered network topologies and configuration variants which were obtained for a random load pattern under the topology-specific deterministic routing schemes described in section 2. These results represent the fundamental performance characteristics of networks for universal use. In sections 4 and 5 we will then focus on the performance effects of specific load patterns with sustained systematic contention and the use of non-deterministic routing schemes.

PERFORMANCE PREDICTION

In the random load pattern used for this investigation, all terminal links are permanently feeding packets into the network at a fixed production rate and with randomly determined destinations from the entire address space. The performance criteria for our evaluation were the maximum sustained throughput (per port) which corresponds to the maximum production rate the network is able to serve without constantly increasing the input queues, and the average packet delay occurring for a given fixed production rate.

The network size was varied in the range of 32 to 512 terminal links. In order to evaluate the performance scalability with network size, we have to consider scaling classes for each topology which are determined by specific configuration parameters. For the Clos networks it seems natural to fix the link ratio and scale the size through the number of switches per stage (with multiple links where possible) and through the

number of stages. The link ratio (16:16, 8:24 etc.) therefore determines a scaling class. For the direct grid or cube networks, scaling classes are defined by a fixed node size a (the number of terminal links per switch, which corresponds to the number a of terminal links per first stage switch in the Clos network). Grids are then scaled through their linear length in each dimension, with a fixed node degree and equal fixed link width per edge, while cubes are scaled through an increasing node degree. Due to our restriction to configurations where all links per switch are utilized, the link width of cubes must be reduced with the growing dimension. The standard scaling scheme for cubes (used in most technical implementations) would add edges of equal link width for each dimension. In order to achieve this with given switches of fixed degree, the link width must be determined by the maximum dimension, and in smaller configurations the unused dimensions are left free. Hence, our scaling scheme achieves better performance for the small configurations by utilizing these free links to extend the link width.

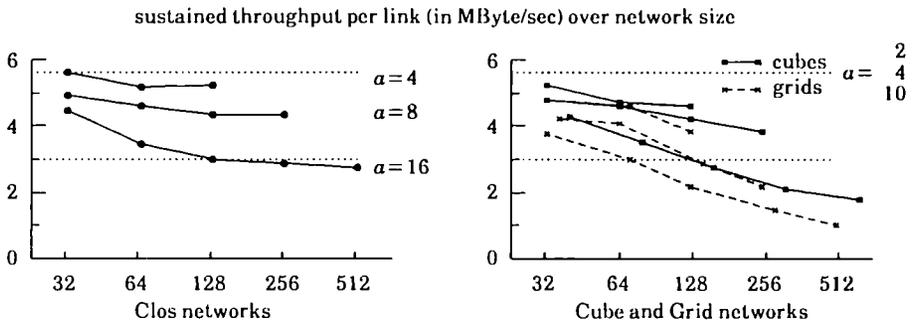


Fig. 3: Network throughput for sustained random load and various network configurations

Fig. 3 shows the maximum sustained throughput per link for different scaling classes of multistage and direct networks over a range of 32 to 512 ports, determined by simulation of the random load pattern. It can be seen that the throughput values vary between 1 and 5.6 MBytes/s depending on the configuration parameters and the network size. The wide difference from the nominal link bandwidth of 10 MBytes/s is partly due to protocol overheads and random destination conflicts in the load pattern, this can be seen from the theoretical full-crossbar throughput at 5.6 MBytes/s. Only further deviations from this level can be attributed to internal contention. It is obvious that the more expensive networks, using more switches and links for the same network size, achieve higher sustained throughput. With respect to performance scalability with network size, it can be observed that the multistage networks (particularly for the expensive 8:24 and 4:28 classes) decrease only slightly in their throughput per link, while the direct networks show much steeper slopes in their throughput curves.

This effect can be explained by the intuitive perception, that constant throughput per terminal link with increasing network size requires a super-proportional increase of network resources (switches and links) to compensate for both, the higher load and the higher diameter of the network. The Clos networks with a link ratio of 16:16 or less do scale their switch number in this way, i.e. proportional to the network size for a fixed number of stages and with an additional logarithmic factor for each additional stage. The cubes and grids in our scaling scheme, however, add switches and links only in a fixed proportion with terminal links and thus do not compensate for the increasing path length. Since the path length grows faster for the grids than for the cubes, the grids'

throughput also decreases faster with the network size. The alternative scaling scheme for cubes indicated above would increase the number of (utilized) links with the diameter as required and thus achieve better performance scaling, but this would change the curves of Fig. 3 by degrading their left wing rather than upgrading the right one.

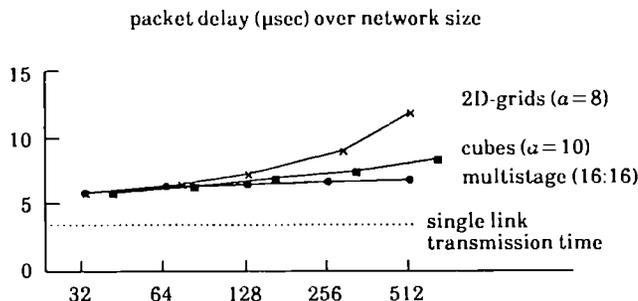


Fig. 4: Average packet delay for sustained random load at 1 MByte/s

Fig. 4 shows the average packet delay for random load at a moderate production rate of 1 MByte/s. This includes the transmission time of a 32-byte packet over the 10 MByte/s links. It can be observed that the delay increases very slowly with network size for multistage and cube networks, where it remains below $8 \mu\text{s}$ for up to 512 nodes (i.e. at a total network throughput of 0.5 GByte/s), while it increases much faster for the low-dimensional grids. A fast increase of packet delay occurs in particular, when the load production rate approaches or surpasses the maximum sustained throughput (i.e. at network saturation), which is the case for the large, cheap configurations with a limit throughput of 1 MByte/s or below. Investigations of various networks at a load production rate corresponding to their respective maximum sustained throughput resulted in a packet delay of about $12 \mu\text{s}$ (for 512 ports) independently of the network topology and configuration.

COST MODELLING

The results presented so far allowed a performance prediction in absolute measures for the individual networks and a restricted comparison within each topology class. For a performance comparison between different topologies, in particular between multistage and direct networks, we need a fixed relation between the corresponding configuration parameters. Such a relation is defined in a natural way by comparing network cost. Knowing cost and performance of a network configuration (or scaling class), the better network is one with higher performance at equal cost or with equal performance at lower cost (for a given size, or over a given size range).

As an initial approximation for a reasonable cost model we can take the number of switches divided by the network size, assuming that all links provided by the switches are actually utilized and that the wiring cost between the switches are significantly lower than the switch costs (which is a realistic assumption for small to medium size networks). Since the number of switches in multistage networks is proportional to the network size for a fixed number of stages, we can determine the scaling classes of cube (or grid) networks with an equal cost ratio. The cost-equivalent configuration parameters are listed in Tab. 2 (link ratio for the 3-stage networks, link ratio combinations first stage/center stage for the 5-stage networks, node sizes for the cubes). The limit size sets the limit where the multistage networks have to add an extra stage and hence

increase their cost ratio.

Limit Size	Clos network configurations	cost equivalent cube configurations
512	16:16	$a = 10..11$
256	8:24	$a = 4..5$
128	4:28	$a = 2$
8192	16:16 / 16:16	$a = 6..7$
2048	8:24 / 8:24	$a = 1$
4096	8:24 / 16:16	$a = 2..3$
2048	4:28 / 16:16	$a = 1$

Tab. 2: Cost-equivalent network configurations

Looking back to the performance curves of Fig. 3 and comparing Clos and cube curves of equivalent cost (i.e. 8:24 Clos with $a = 4$ cubes for up to 256 ports and 16:16 Clos with $a = 10$ cubes for up to 512 ports), it can be seen that the Clos networks have a higher throughput for the larger networks, while the cubes are slightly better for the smaller networks. Taking into account the more realistic scaling scheme with a performance penalty for low-dimension cubes, this presents a performance advantage of Clos networks over cubes at equal costs.

Recent models for a fairer comparison of different network topologies (in particular cubes and grids) do consider the amount of wiring in a network as an important cost factor. Taking the bisection width of a network as the prevailing cost factor, Dally (1) proves a performance advantage of low-dimensional grids over cubes, by compensating the low dimension of the grids with an increased link width. This result can not be simply transferred to the conditions of transputer networks, because Dally's main assumptions do not hold: switch delay is much higher than a single channel cycle and channel bandwidth is not increased with the link width. Also, the given switch types with a fixed number of serial links cannot provide an equivalent bisection width for large grids, because this would require to extend the link width with the grid size.

In our own extended cost model, we have therefore included the wiring cost according to the given conditions and do now consider the number of (utilized) links between system partitions (*clusters*) of equal size. Taking into account that wiring cost are mainly due to the required cables and connectors between distant clusters, we may specify a cluster size of 128 transputers or up to 512 terminal links for which the internal wiring does not significantly contribute to the network cost. Calculating the number of link connections between such clusters in larger configurations, we obtain a substantial cost advantage for 2D-grids and a disadvantage for the Clos networks compared with the cubes. Combining the switch and cabling cost in a single model and assuming a relation of 1:10 for the relative cost of a link cable (with connectors) and a switch component, however, we obtain the result that the switch costs still dominate the total network cost up to a range of about thousand ports. Hence, the cost/performance relations presented so far are not significantly changed by including the cable cost.

The application of our extended cost model gave us the opportunity to determine the cost-optimal configuration parameters for the different topology classes. The diagrams in Fig. 5 show the throughput/cost ratio for varying configuration parameters of multistage and cube networks of different sizes. Although the maximum values cannot be exactly determined from the rather rough curves, it is apparent that the Clos networks have an optimal configuration around $a = 16$ and the cubes around $a = 12$. In

addition, it can be observed that for equal network size the Clos networks achieve better throughput/cost ratios than the cubes. This does still hold, when comparing the throughput/cost ratios for networks of equal size and equivalent cost (according to Tab. 2).

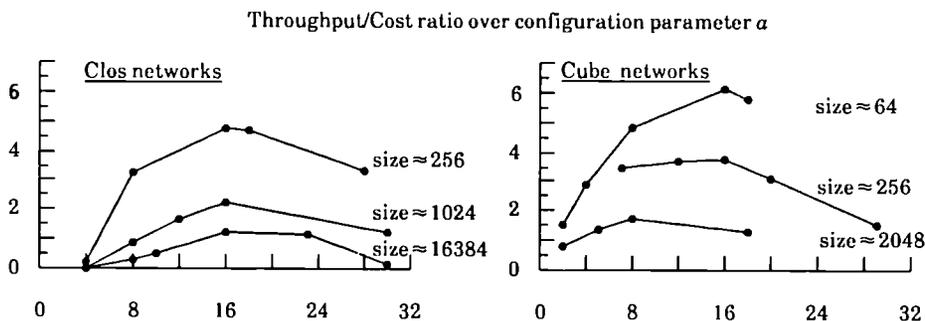


Fig. 5: Cost-optimal configuration parameters for multistage and cube networks

From the evaluation of the network topologies and configurations for sustained random load, it can therefore be concluded that the multistage networks are slightly superior to the binary n -cubes with respect to performance, cost/performance ratio and scaling behaviour, and both are clearly superior to low-dimensional grids. In the following sections, we will point out further and even more significant advantages of the multistage network class for critical systematic load patterns.

4. PERFORMANCE GAINS THROUGH NON-DETERMINISTIC ROUTING

The random load patterns considered so far did expose the effects of non-systematic contention under constant high traffic conditions, which is a major performance characteristic determining the communication behaviour in many other load cases as well. Real applications do also frequently involve regular communication patterns which lead to systematic bottlenecks (hot spots) at certain links in the network. In such cases, packets are routed to the hot spot faster than they can be served by the destination link, and thus a saturation tree emerges. This is a well-known effect in the case of external hot-spots (destination conflicts), which are discussed in more detail in section 5. A similar effect, however, does arise when packets for different destinations must be routed through the same internal channel due to the pre-determined routing scheme, although other possible routes remain idle. In principle, this situation can only be improved by permitting various routes for the same sender/receiver pair, i.e. by non-deterministic routing schemes which dynamically utilize the available resources of the network in a uniform way.

One such approach is the *two-phase routing* or *universal routing* method (8) where each packet is first routed to some randomly determined intermediate node before it is forwarded to its original destination. In this way, any systematic routing pattern is destroyed and internal hot-spots cannot occur. Internal contention is not completely prevented, but it is equally distributed over the entire network which thus behaves like in the case of random load. Since each packet must be routed through the network twice, the total network load is doubled and so is the packet delay for communication patterns without internal contention or with random behaviour. Hence, two-phase

routing trades in best case performance for improved worst case performance in order to achieve a better balanced and more predictable behaviour for arbitrary load.

Two-phase routing is supported by the C104 switches through their capability to generate random address headers for passing packets and to strip off such headers on reaching the intermediate destination (5). In direct networks such as grids or cubes, it is necessary to use separate sets of links for each routing phase in order to avoid deadlock. The duplicated links also provide the sufficient bandwidth to compensate for the extended path length, so that a desired throughput level can be maintained. In the Clos networks, the desired effect of an equal distribution of all packets over all links in the network to minimize internal contention can be more easily achieved by routing randomly from the network entry to the center stage and continuing deterministically from there. This corresponds to the two phases of the previous scheme, but maintains the number and sequence of routing stages and hence, does not require duplicated links to prevent deadlock. The path length each packet has to pass is the number of (logical) stages, which is a slight increase compared with the deterministic routing scheme where some packets may take a shortcut without passing the center stage.

From the simple and favourable application of random routing to multistage networks we derived a further optimization of the routing scheme which maintains the good properties of random routing (for the Clos networks), but prevents random contention and utilizes shortcuts. It is a locally adaptive scheme where the route towards the center stage is selected randomly among the idle links only (when all links are busy, one of them is selected arbitrarily). For easier implementation the random selection can be approximated by a round-robin selection scheme. In certain cases of permutation load which lead to maximum contention with the deterministic routing, the adaptive scheme is able to completely resolve the contention and achieve the maximum possible link throughput (2). As a consequence of the approving results, the proposed *group adaptive routing* mechanism has been included into the specifications of the C104 packet router component.

It should be noted that the adaptive routing scheme can also be exploited in direct networks by selecting adaptively among the multiple links of the edge determined by the routing algorithm. Since the link width in reasonable grid or cube configurations is restricted to small numbers, the advantage gained from the dynamic adaption is less than for the Clos networks where it can be chosen between 16 or even 24 center links.

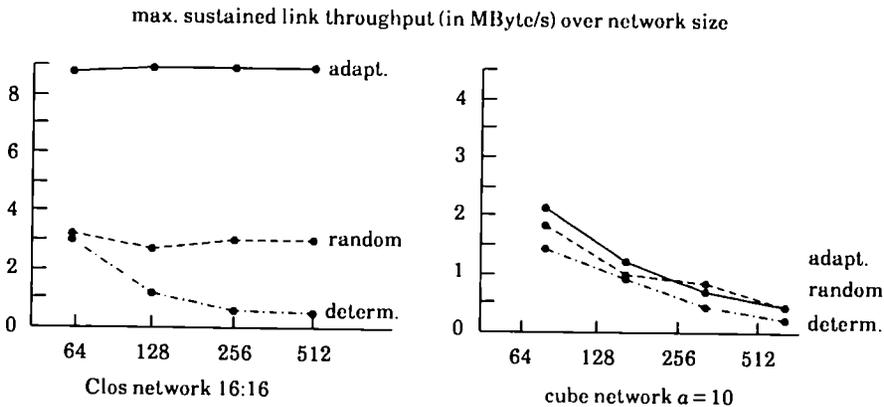


Fig. 6: Performance gain of non-deterministic routing schemes for worst-case permutation load

Fig. 6 shows the effect of random and adaptive routing on the maximum link throughput for a sustained permutation load pattern which is chosen (depending on the network topology) to produce the maximum contention in case of deterministic routing. Since there are no destination conflicts, the maximum obtainable throughput is much higher than for random load, and it is actually achieved (for this special permutation) with the adaptive routing scheme. Multistage random routing leads to a substantial advantage over the deterministic routing, but of course admits conflicts in each stage. For the cube network (of comparable cost) the two-phase random routing scheme does improve performance, but only to a lesser amount (note the extended throughput scale). This is due to the fact that only half of the link width can be used in each phase while the deterministic (and adaptive) routing schemes can utilize all links on a single pass through the network. Adaptive routing therefore surpasses the random scheme slightly, but cannot achieve comparably large throughput gains as in the Clos networks, because the link groups for adaptive selection are smaller.

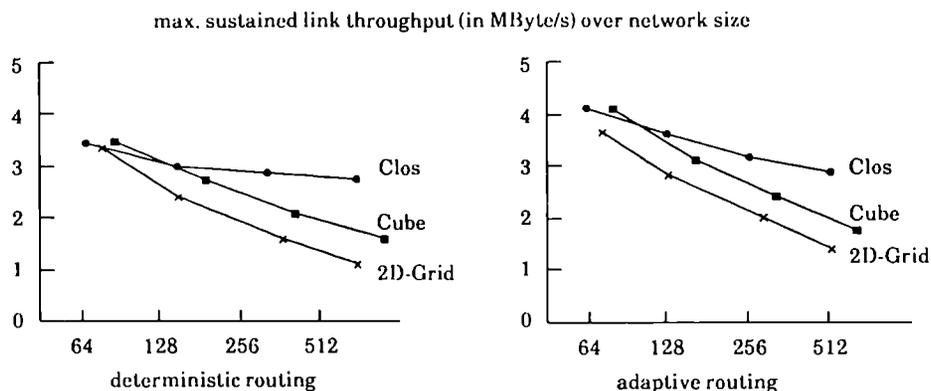


Fig. 7: Performance gain of adaptive routing for sustained random load

In Fig. 7 the effect of adaptive routing on the random load pattern of section 4 is demonstrated. It must be noted that a random load cannot be expected to be distributed in a more balanced way by any non-deterministic routing scheme. Two-phase random routing would therefore on the contrary lead to significantly reduced throughput due to the doubled path length. As indicated above, even for the simpler random routing scheme in the Clos networks the average path length is slightly increased and therefore a small throughput reduction must be expected for random load. Adaptive routing, on the other hand, still improves the sustained random throughput by reducing the amount of random contention. This improvement amounts to about 5-30% for all networks, where the higher gains for the smaller Clos networks arise from the fact that multiple links can be used adaptively on the return path from the center stage to the terminal stage, while the full-blown 3-stage configuration has only one link between each pair of switches. In contrast to this effect which seems to increase the performance reduction with growing network size, our analytical investigations for long-range scalability proved a slight advantage of adaptive routing for the scaling of throughput with increasing number of stages (2) (31 instead of 44% throughput reduction per terminal for the network size growing from 100 to 16000 ports).

5. PARTICULAR PERFORMANCE EFFECTS

While internal hot-spots can well be resolved by random or adaptive routing, this is not possible for external hot-spots caused by systematic destination conflicts in the load pattern. The adverse performance effects of even a small proportion of such hot traffic in otherwise random load have been extensively investigated and discussed in (6). To complete the performance predictions and comparisons of transputer networks for universal use, we also had to investigate our network models for their hot-spot behaviour. This was done with the following load model: All network ports were charged with a sustained random load at a fixed production rate of 2 MByte/s. For a short period (the hot phase) a variable proportion of packets (the hot packets) was directed to a fixed destination (the hot spot).

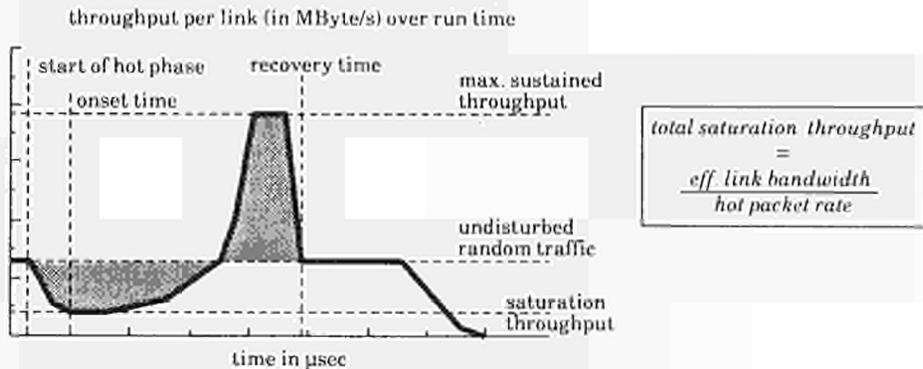


Fig. 8: Typical Throughput profile for hot-spot load

In Fig. 8 we show the typical throughput profile for this load, which is characterized by few parameters. During the hot phase a saturation tree develops reducing the throughput from the random to the saturation level. Our investigations proved that the total saturation throughput depends directly on the proportion of hot packets (the hot rate) and is well approximated by the relation shown in Fig. 8. In particular, it does not depend on the network topology, configuration and routing scheme. The saturation level is maintained throughout the hot phase and after its end until the hot rate ejected from the senders' queues (which have filled up during the hot phase) begins to decrease. Then the average link throughput rises and reaches the maximum sustained random throughput (determined in the previous sections) when the hot rate out of the queues is zero, and random packets are fed into the links at the maximum rate until the queues are emptied. Finally the throughput decreases again to the original level corresponding to the random production rate.

The recovery time from the beginning of the hot phase until complete disappearance of the saturation tree (when the throughput has levelled off again) depends on the length of the senders' queues (which was infinite in our model and thus determined by the duration of the hot phase), but also on the maximum throughput level achieved during recovery. This leads to a slight performance advantage of networks with better sustained random performance, e.g. Clos networks with adaptive routing. In general, however, the performance differences between the considered networks remained rather low, and the recovery times which were in the millisecond range even for very short hot phases could not be significantly reduced by structural measures with the given components.

Since these are results for a global hot spot where all nodes contribute equally to the hot rate, the question arises whether a more local hot-spot pattern affects the throughput in other regions of the same network. We therefore started an investigation of partitioned systems with individual load patterns per partition.

A standard approach for network partitioning, particularly for multi-user operation in a space-sharing environment, is the recursive decomposition into sub-networks of equal structure, e.g. by reiterated bisectioning. Defining "clusters" as groups of terminal ports connected to a single switch, it is obviously desirable to include only entire clusters (or groups of those) into the partitions, because a larger proportion of traffic remains internal to the clusters and can thus be handled faster, with no contention except destination conflicts and clearly separated from other partitions.

For the Clos networks with their standard labelling scheme defined in section 2 this does, however, imply that only part of the center links are utilized (depending on the partition size). This leads to decreasing throughput with decreasing partition size, as can be seen in Fig. 9a, where sustained random load at maximum rate was applied within each partition. Only for partition size 1, where the entire partition is connected to the same switch and no packets have to flow through the center links, the maximum throughput is obtained. In order to avoid that effect, the labelling scheme must be optimized according to the partitioning. As in the case of random or adaptive routing, this leads to constantly increasing throughput with falling partition size.

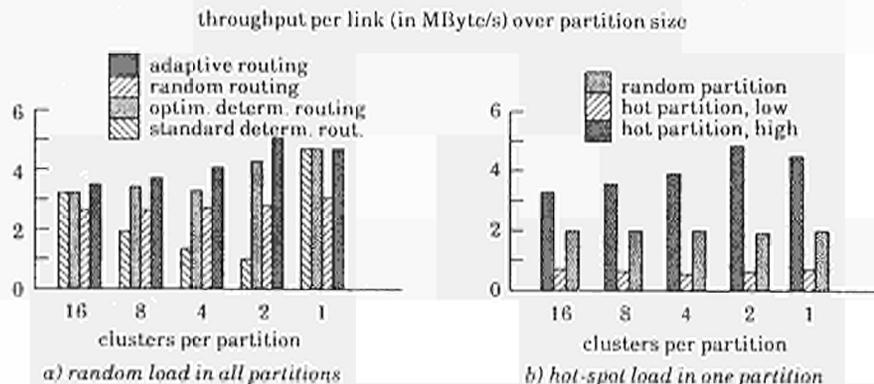


Fig. 9: Performance comparison for various partitionings of a 256-port Clos network

Fixing the sustained random load in all partitions at 2 MByte/s and changing the load pattern in one partition to the hot-spot pattern described above, results in the throughput behaviour shown in Fig. 9b (for the case of adaptive routing). The number and frequency of hot packets generated in the hot partition was fixed for all partition sizes in this investigation, thus obtaining a growing hot rate with decreasing partition size. This increase of the hot rate leads to a corresponding reduction of the saturation throughput (according to the relation shown in Fig. 8), which is compensated by the decreasing number of links per partition and hence, we observe a constant saturation throughput per link for all partition sizes. The maximum throughput for the hot partition during the recovery phase corresponds exactly with the sustained random throughput (for adaptive routing) shown in Fig. 9a. The same effect occurs for any other routing scheme, and it presents the only significant difference between the routing schemes in this investigation. The performance in the random partitions is not affected by the hot-spot load in the other partition, because the packets of different partitions are strictly

separated on different links (although passing through the same center stage switches). This is a result of the partitioning scheme which is also obtained for random or deterministic routing.

Finally, it should be noted that also for the direct networks a full performance separation between partitions is achieved by the standard decomposition into sub-cubes and -grids, because the labelling schemes are such that packets always remain in the sub-network spanned by their source and destination ports. For the random routing scheme it is necessary (in contrast to the Clos network) to redefine the random intervals in order to restrict the intermediate destination to the same partition.

6. CONCLUSION

We have presented a wide range of investigation results for the most relevant subset of network topologies and configurations for scalable, universal, transputer-based systems. It could be proven that transputer networks with C104 switches are able to achieve impressive performance at reasonable cost, e.g. more than 4 MByte/s sustained random throughput per port up to 1000 network ports with less than 15 us average packet delay can be obtained at the cost of less than one switch per port. With the non-deterministic routing mechanisms and in particular, the newly developed adaptive routing scheme, it is possible to resolve systematic internal contention effectively and thus achieve good and well-scalable performance for realistic communication patterns without the need for a topology-specific application mapping.

As a main conclusion for C104-based networks up to the range of 1000 network ports we recommend the multistage networks of the Clos-type to be the best choice for several reasons:

- they make the most efficient use of the non-deterministic routing mechanisms, in particular adaptive routing,
- they are the least expensive topology when built in their most cost-effective configuration (16:16 link ratio),
- in any configuration they achieve the best performance at given cost and have the lowest cost to obtain a required performance level,
- they have a (slight) performance advantage in case of external hot-spots (through their better maximum throughput) and can easily be partitioned into performance-isolated sections.

The investigations leading to these results were carried out in close cooperation with the project partners who provided the necessary specifications for the component models and load patterns and took advantage of the resulting performance predictions for their own optimization of communication mechanisms and strategies. The conclusions presented above have directly influenced the design of the C104 component and they represent a basis for system design decisions in the follow-up project GP MIMD (EP 5404).

ACKNOWLEDGEMENTS

This work has partly been supported by the EEC under ESPRIT project EP2701 (PUMA). The author would like to express his special thanks to Holm Hofestaedt and Erwin Reyzl, who performed most of the investigations reported in this paper and helped improving and revising its contents. Thanks are also due to Peter Thompson of Inmos

and Denis Nicole of Southampton University for very helpful and enlightening discussions on the issues and results of our work.

REFERENCES

- (1) W.J. DALLY: Performance Analysis of k-ary n-cube Interconnection Networks; IEEE Trans. Comput., 6 (C-39) 1990, pp. 775-785
- (2) H. HOFESTAEDT, A. KLEIN, E. REYZL: Performance Benefits from Locally Adaptive Interval Routing in Dynamically Switched Interconnection Networks; Proc. 2nd Europ. Conf. on Distr. Memory Computing (EDMCC2), Munich, April 1991, pp. 193-202
- (3) INMOS Ltd.: The T9000 Transputer Products Overview Manual; Inmos Databook Series, 1991
- (4) J. VAN LEEUWEN, R.B. TAN: Interval Routing; The Computer Journal 30 (4) 1987, pp. 298-307
- (5) D. MAY, P. THOMPSON: Transputers and Routers: Components for Concurrent Machines; Proc. Japanese Occam User Group, 1990
- (6) G.F. PFISTER, V.A. NORTON: Hot Spot Contention and Combining in Multistage Interconnection Networks; IEEE Trans. Comput., 10 (C-34) 1985, pp. 943-948
- (7) E. REYZL, H. ECKARDT: Performance Evaluation for High-Performance Interconnection Networks; Proc. 11. ITG/GI-Fachtagung "Architektur und Betrieb von Rechensystemen", Munich, March 1990, pp. 275-287
- (8) L.G. VALIANT: A scheme for fast parallel communication; SIAM J. on Computing, 11, 1982, pp. 350-361

Using the Genesis Distributed Memory Benchmarks to evaluate the SUPRENUM Computer

Cliff Addison
Centre for Mathematical Software Research
University of Liverpool
Liverpool, U.K.

Nigel Bishop¹, Tony Hey, Roger Hockney, Ivan Wolton
Department of Electronics and Computer Science
University of Southampton, U.K.

Abstract:

The Genesis Distributed Memory Benchmarks represent a significant step forward in the evaluation of Distributed Memory MIMD systems. The initial version of this benchmark suite, described in [1], was augmented and modified in order to evaluate the SUPRENUM computer. This evaluation was then used to estimate performance on the proposed Genesis machine.

The benchmark suite is briefly described along with the methodology employed in obtaining and presenting results. The SUPRENUM computer is then described, along with some general results of the evaluation and then selected benchmarks are described in detail. The paper concludes with some observations relating to the challenges of providing good benchmarks for distributed memory computers and the steps being taken to address these challenges in the Genesis Distributed Memory Benchmarks.

1. Introduction

During the Esprit Project 2702 - Genesis, a number of benchmark programs were developed to evaluate the performance of the German SUPRENUM distributed-memory (DM) computer compared to other parallel computers (including multiple-vector and shared memory architectures). Preliminary results for the so-called Genesis pre-study have already been published in outline [1]. In this paper, the refinement of this benchmark suite to assist in the SUPRENUM evaluation is discussed, some general results provided and selected benchmarks are described and analysed in more detail.

The Genesis benchmarks are required to be written in standard Fortran-77 where possible, with both a sequential (base) version and distributed memory version being supplied. The normal arithmetic precision is 64-bits, although 32-bit codes may be included if there is good reason. Several of the benchmarks also include a version with explicit vector instructions expressed in Fortran-90 syntax [2]. This version was important for the SUPRENUM computer and will be of further use in benchmarking SIMD computers, such as the Connection machine, that have adopted this language, and for comparing the performance of different Fortran-90 compilers as they become available. Optimisation of the benchmark codes is permitted - indeed encouraging in order to show the full potential of the hardware.

¹ Department of Computational and Applied Mathematics, Centre for Non-linear Studies, University of the Witwatersrand, Johannesburg 2050, South Africa

The Fortran-77 sequential version of the benchmark defines the problem and algorithm to be used, and with minor modifications and tuning should run on vector computers and shared-memory (SM) parallel computers, using the currently available auto-vectorising and auto-parallelising (or auto-tasking) compilers. The distributed-memory version of the benchmark, on the other hand is a fundamentally different program in which the work to be performed is explicitly distributed to multiple processors which may only communicate with each other by sending messages over communication channels using I/O (send/receive) statements. The translation of the sequential version to an efficient DM version without the help of software tools is usually a non-trivial task measured in man-months of effort. Currently a number of automatic software tools are being developed to ease this task, and it is expected that these will significantly reduce the labour involved. Nevertheless, significant human programming effort is still likely to be required. The value of the Genesis Benchmarks is that they reduce this work by providing distributed-memory message-passing (DMMP) programs for a number of important problems in physics.

Another important aspect of the Genesis Benchmarks is that they provide an approximate timing model which allows the variation of performance with both the problem size and the number of processors to be estimated. It is believed that this feature of the benchmarks will prove particularly valuable for predicting the performance of future computers, and in understanding how their performance scales.

A reader might well ask what is the justification for introducing yet another suite of benchmarks - after all the PERFECT Club [3], SPEC [4] and Euroben [5] have already established substantial and growing suites of programs covering similar scientific applications. The reason is that none of the benchmarks mentioned above yet provide DMMP programs. Their programs are sequential and suitable for testing single CPUs, or shared-memory multiple CPUs through the use of auto-tasking. They cannot be run on DMMP computers. The only other DM benchmarks known to us are those from the CalTech Performance Evaluation Project (CPEP [6]), which indeed were assembled at about the same time as the Genesis benchmarks and for similar reasons. Our work therefore has some overlap with that of CPEP. Their work covers, at present, a much wider range of computers, but does not provide such a full timing analysis as that given here. At the present early state in the evolution of DM benchmarks we feel there is justification for this and other independent initiatives.

This particular paper focuses on the influences the SUPRENUM evaluation had on the Genesis distributed memory benchmarks. In addition, it is intended to publish the results from these benchmarks as a series of papers entitled "The Genesis Distributed Memory Benchmarks I, II etc.", with a subtitle giving the computers and benchmarks included.

2. Problems selected

The evaluation of the SUPRENUM computer started with a series of monthly meetings of the Genesis Evaluation sub-group² to discuss the benchmarks that were to be used and the methodology that would be employed.

² This sub-group included scientists from: SUPRENUM GmbH, GMD, NAG Ltd, CHAM Ltd, European Centre for Medium Range Weather Forecasts and Universitat Politècnica de Catalunya as well as scientists from the University of Liverpool and Southampton.

One of the first results of this collaboration was to identify the broad classes of benchmarks required in order to present a comprehensive view of the available performance. These include:

- Benchmarks that can exploit almost arbitrary amounts of both fine-grain and coarse-grain parallelism.
- Benchmarks that have proven performance on large, vector machines with small numbers of processors.
- Benchmarks that can exploit large amounts of coarse-grain parallelism but have limited use for fine-grain parallelism.
- Computationally intense benchmarks with a controllable level of interprocessor communications.
- Computationally intense benchmarks with a controllable amount of disk I/O
- Benchmarks that have potentially poor vector / parallel performance
- Benchmarks that involve extensive communications and data movement with relatively little computation.
- Single node benchmarks that can exploit fine-grain parallelism.

All of the above classes of benchmarks were covered by the Genesis benchmarks, although time and system difficulties prevented extending certain benchmarks to perform disk I/O.

Consistent with the above requirements, the problems and algorithms selected for the distributed memory benchmarks range from kernel measurements to full applications and were partly conditioned by what was available to the researchers involved. As a consequence, they are primarily research programs developed at Universities for applications in physics and chemistry.

The benchmarks used in the evaluation are given below. Benchmarks marked with an asterisk(*) are not included in the present version of the Genesis benchmarks.

1. Single node benchmarks and synthetic code fragments

- BLAS 1, 2 and 3
- * LAPACK
- LINPACK benchmarks
- * NASA benchmark kernels and Livermore loops
- Single node FFTs
- Inter-node communication
- Matrix transpose

2. Parallel numerical kernels

- parallel LU factorisation, triangular solve and matrix-vector product
- Matrix multiply
- QR decomposition using Givens or Householder transformations
- 1-D parallel FFTs
- 2-D Red-Black Relaxation for Poisson's equation
- 2-D multigrid Poisson solver
- CG (Conjugate Gradient) solver for QCD

3. Large applications or kernels thereof

- Meteorology: Pseudo-spectral and multigrid Helmholtz solvers
- * Meteorology: A finite difference model for the shallow water problem
- QCD (Quantum Chromodynamics): the kernel of a lattice gauge theory Monte Carlo simulation
- TOMCATV (a mesh generation program)
- Molecular dynamics
- General relativity

3. Methodology

Two fundamental reasons for benchmarking are:

- A desire to compare performance of benchmarks on different machines.
- A desire to form and validate timing models as a prediction tool.

The former is an extremely thorny problem to handle properly, particularly because the basic characteristics of the different DMMP computers differ widely. For instance, code that would deliver nearly optimal performance on a transputer array might deliver poor performance on SUPRENUM or the Intel iPSC/2. Similarly, if code written for the SUPRENUM exploited a complicated communications topology, performance on arrays of T8 transputers might be disappointing. As a result, the benchmarks need to be written for a generic programming model, which encapsulates the critical features of current and future DMMP computers.

As the Genesis Benchmarks are modified over time, it is hoped that it will become increasingly easier to obtain good performance from a wider range of DMMP computers.

The other desire pertains to performance prediction. A benchmark suite should reveal the weakest points of a computer's design. It is important to then be able to estimate performance if underlying characteristics are changed to overcome these weaknesses. In effect, we need to know how sensitive the benchmarks are to changes in hardware characteristics. It is also desirable to estimate performance on machines that have not yet been built. The need for timing models for this prediction focused attention on what aspects of a benchmark should be timed and how.

The following guidelines related to this were agreed upon in the evaluation:

- Benchmarks generally should be run on a quiet system with only one user; this includes the host machine if communication with the host is a part of the benchmark. The equivalent of elapsed time should be measured.
- When a benchmark represents only a portion of an application code, task creation time must not be included in overall benchmark times. In this case, the total time is defined to be the maximum processor time measured after an initial synchronisation of all tasks.
- The available clock is often only accurate to a fifth of a millisecond. This means that small events cannot be measured directly. If small events must be timed, then there must be a separate benchmark for these small events that can be run together with the main benchmark.

- Benchmarks should not only measure the time of major modules, but whenever possible, they should also give the number of "useful" floating point operations³ (written flop) performed within that time as well. Mega-flop rates (written Mflop/s) can then be determined, which help to measure processor utilisation. Times should be presented along with Mflop/s because the latter alone are open to abuse too easily.
- Test results must be reproducible so that different evaluators can produce similar results. This has certain implications for routines that use random number generators.
- Two general absolute measures of performance will be used: the Temporal performance (defined as the inverse of the elapsed time), and the Benchmark performance (defined as the fixed standard floating-point operation count for the benchmark, divided by the elapsed time) The performance variation is studied as the number of processors increases with a fixed problem size, and as the problem size increases for a fixed number of processors Scale-up, the performance when the problem size is scaled proportionately to the number of processors is of secondary importance and can be computed from sets of runs where either the problem size or number of processors is fixed.

The presentation of performance in terms of relative measures, such as speed-up or efficiency, was strongly discouraged as these can obscure detail.

It was felt that both single node and parallel versions of a benchmark were necessary, even when realistically sized problems could not be run on a single node. The same basic algorithms were to be employed in both, although different optimisations might have been used.

The algorithms for benchmarks are often selected for their parallel potential and may not be widely used in single processor environments. In such cases it was agreed that "credibility runs" should be included, which consist of running a single processor version of the relevant benchmark and a compatible version of the "best" single processor algorithm on the same test problems can be used. Examples of suitable processors for such runs would be single processors of a Cray XMP or an IBM 3090 - machines that have vector processors and a relatively large amount of main memory.

Several other aspects of the evaluation methodology benefited significantly from the work of the PERFECT Club:

As mentioned earlier, optimisation is actively encouraged, but the optimisations (and the resulting changes in performance) should be recorded. The idea of recording the time each optimisation took to implement was viewed as desirable, but impractical.

Optimisations may alter a program's performance so that similar, but not identical, results may be obtained. It is therefore important to have some easy way for a tester, who is not the developer, to see if a version is acceptably accurate. If each benchmark has a small set (say fewer than 10) of values that can be compared among different versions of the program, then a reliability test can be made automatically by computing the relative difference of these values when compared with a baseline version. A benchmark run would be acceptable only if these differences were all smaller than pre-defined thresholds.

Benchmarks must be specified by a low dimensional parameter space. This is necessary to obtain a complete picture of a benchmark's performance on a target machine with a 'small' number of test runs. A benchmark would be run at predetermined values within this parameter space in order to obtain a clear picture of how performance

³ For instance, redundant floating point operations performed to avoid communications should not be included.

changes with varying parameters (such as problem size or number of nodes). As part of the specification, the developer must also indicate the legal ranges for these input parameters, including how these parameter ranges are inter-related.

Standards pertaining to lower level evaluation details have also been established, such as the availability of machine readable timings to a particular number of significant figures, the use of particular performance metrics for comparisons and graphical display standards. The intention is that the results are presented in a form that allows readers to examine the data and make their own interpretations.

4. General result for SUPRENUM

The SUPRENUM computer was the culmination of a multi-year collaboration of several leading German institutions[7]. The machine has a hierarchical structure consisting of up to 16 clusters connected by token ring networks, where each cluster consists of 16 vector nodes (20 Mflop/s peak chained performance with 8 MBytes of memory), a high-speed cluster disk and communication nodes. Communication within a cluster is over a highspeed (320 MByte/s peak) bus. The system included a sophisticated Fortran compiler with extensive and easy to use extensions for communications as well as support for the Fortran-90 array constructs.

Amongst other things, the evaluation sub-group of Genesis was requested to verify that SUPRENUM is a distributed memory supercomputer and to identify the strengths and weaknesses of the SUPRENUM system.

Despite severe difficulties with late hardware delivery, unstable systems software, faulty compilers, personnel recruitment and short time frames this was accomplished. The results were not as extensive as had been planned, but it was still possible to provide some significant insight into what SUPRENUM could and could not provide.

It is possible to obtain nearly ideal performance with a single SUPRENUM node, but only on blocks of long, dense vector operations. For example, matrix multiplication written in Fortran attained 14.6 Mflop/s on 400 matrices. Several other Fortran benchmarks attained good performance, but only on large problems. For example, the hand optimised LU decomposition of a 700 by 700 matrix attained about 6.5 Mflop/s, but on 100 by 100 matrices, the rate was only just over 3 Mflop/s. A radix 2, complex FFT attained a performance of 3.7 Mflop/s on 32 FFTs, each 1024 words long.

The long start-up time for vector operations was a significant shortcoming. Another major difficulty was the large gap between vector and scalar performance on a single node. This meant that non-vectorisable code could only attain a mediocre performance, even if it parallelised very well. There were also a number of subtle optimisation issues that had to be handled carefully. While the group attempted to spread the knowledge of these as widely as possible, the large number of part time benchmarkers meant that the information was not always in the right hands at the right time.

Measured results on parallel systems have been variable, partially reflecting the effort (and sometimes expert knowledge) required for optimisation. The highest performance so far has been on matrix multiplication. On 64 nodes, 870 Mflop/s were achieved on optimally sized (but large) matrices. The multiplication of two 3200 X 3200 matrices has been measured at 776 Mflop/s. These figures correspond to 68% and 61% parallel efficiency respectively.

Several results suggest that SUPRENUM can provide supercomputer performance if the application vectorises well and can utilise a coarse level of parallelism. For example:

A highly optimised code for quantum chromodynamics (QCD) has been developed. This code contains the equivalent of the QCD benchmark kernel used in the PERFECT Club ([3]). A performance of over 200 Mflop/s has been measured on 64 nodes. When run on the PERFECT Club test problem for QCD, the SUPRENUM code's performance is roughly half that of a four processor Cray XMP, one sixth that of an eight processor Cray YMP but nearly twice that of a Cray 2.

Another result is for TOMCATV, which is a mesh generation program that is part of the SPEC benchmark suite ([4]). The code is highly vectorisable and performs little I/O. A performance of 67.5 Mflop/s has been observed on 16 nodes. The SUPRENUM performance was similar to that of a single processor Cray XMP or Cray 2 on similarly sized problems.

A program for the shallow water code was developed as a benchmark program for parallel computers at the NCAR, Boulder. The algorithm is fully vectorisable, using an explicit second order time stepping scheme. The parallelisation is done by grid partitioning. The parallel version of the benchmark had been developed for the iPSC/2 and was ported to SUPRENUM using the iPSC / SUPRENUM compatibility library by Prof. O. McBryan (University of Colorado at Boulder). Preliminary results gave a speed of 4 Mflop/s on 1 node and 48 Mflop/s on 16 nodes when the problem size was scaled.

These were some of the encouraging results. As mentioned above, optimisation was difficult, due in large part to long vector and communication start-up times as well as problems with compilers and the system software. For example, routines that performed communications using the language extensions could not be compiled with the optimisation and vectorisation turned on. This distorted some benchmark results considerably, sometimes suggesting that perfectly reasonable parallel algorithms were inferior to the sequential algorithms.

5. Communications

A fundamental aspect of a multi-node DMMP computer is the latency and speed of communication for messages between different nodes of the multi-node network. The success of a DM computer depends on keeping the latency (message start-up time) low and the speed (asymptotic transfer bandwidth) high. If this is not done, one can easily find that the time of execution is completely dominated by message communication time, and that the expected arithmetic speed is not realised in practice.

Communications were highly sensitive to the type and number of objects being communicated. Best results were obtained for double precision arrays, which was also the most important case.

Several communication benchmarks were written to assess both asynchronous and synchronous communications. When the benchmarks were fully optimised, message start-up times were less than 2ms for synchronous communications and for particular patterns of asynchronous communications. The start-up time for general asynchronous communications. The start-up time for general asynchronous communications was nearer 3 ms. These are the same for communications within a cluster (on the cluster bus) or between clusters (on the SUPRENUM bus).

Within a cluster, the communications bandwidth for a message is around 12 MByte/s with synchronous communications and around 8.6 M Byte/s for asynchronous communications. This is insensitive to the number of nodes simultaneously communicating over the cluster bus.

Communications between clusters had a lower bandwidth, but worse, the bandwidth per message decreased as the number of simultaneous messages between the cluster increased!

One of the benchmarks that measured the communication time between two processors of the same cluster was the "Pingpong Experiment". In this experiment a message of variable length, n , is sent from a master processor to a slave processor. The slave receives the message and immediately returns it to the master. Half the time for this "pingpong" exchange is recorded as the time, t , to send a message of length, n . The time as a function of message length has been fitted by least squares with the parameters $(r_\infty, n_{1/2})$ by the following relationship (see [8]):

$$t = (n + n_{1/2})/r_\infty$$

when the average communication rate is given by

$$r = \frac{r_\infty}{(1 + n_{1/2}/n)}$$

and the start-up time by

$$t_0 = n_{1/2}/r_\infty$$

As mentioned earlier, the asymptotic bandwidth, (r_∞) shows considerable variation on the SUPRENUM, depending on how the data to be transferred is specified in the I/O list of the send statement. A variable length array in Fortran-90 syntax in single precision achieved 0.67 MByte/s in these experiments, whereas the same statement specified in double precision achieved 4.8 MByte/s.

It should be pointed out that these pingpong measurements on asynchronous communications gave a slightly lower start-up time of 2.65 ms (vs. around 3 ms measured in a 4 node ring and in "communication only" versions of several benchmarks), and a lower bandwidth of 4.82 MByte/s (vs. around 8.6 MByte/s measured by other means) This discrepancy still has not been satisfactorily explained, but the difference in performance predicted by these curves grows slowly, so that for modelling purposes either would probably be adequate.

It is informative to compare this SUPRENUM performance to that measured on the Intel iPSC/860 using the same pingpong experiment. For messages less than 100 bytes, the iPSC/860 start-up time is 73 μ s and for longer messages, the rate is 200 μ sec. The bandwidth for the iPSC/860 is 2.8 MByte/s for larger messages. Since the startup time determines the transfer rate for short messages (say < 100 Bytes), we see that the SUPRENUM is 45 times slower than the iPSC/860 for short messages. On the other hand the SUPRENUM has over three times the bandwidth (assuming the most favourable format). From this, we can conclude that the iPSC/860 is faster at transferring messages for all lengths less than around 11,600 Bytes.

The long message start-up times for SUPRENUM stem from a number of factors, including the fact that the designers had written system's routines and had provided language constructs which were flexible and comfortable, but these came at a price. The comparison with the Intel times are therefore slightly unfair. SUPRENUM message start-up is the same for any two processors, while the Intel start-up increases with the distance between processors. In addition, the send and receive operations in SUPRENUM Fortran allow users to send multiple objects in one message with the buffering

performed by the system. Intel users must perform their own buffering into a single array and then pass this to the communications routine.

A fundamental aspect of a multi-node DMMP computer is the latency and speed of communication for messages between different nodes of the multi-node network. The success of a DM computer depends on keeping the latency (message start-up time) low and the speed (asymptotic transfer bandwidth) high. If this is not done, one can easily find that the time of execution is completely dominated by message communication time, and that the expected arithmetic speed is not realised in practice.

6. Transpose

Three variants of parallel matrix transpose were benchmarked:

- GRID runs on an $n \times n$ array of processors, each holding a square sub-matrix. The transpose is performed by processor (i,j) transposing its own sub-matrix internally then sending it to processor (j,i). Pairs of processors communicate with each other, and there is no overall synchronisation of the transpose.
- STRIP runs on n processors, with the matrix split into n columns. Each processor deals with its own column as n sub-matrices, transposing these sub-matrices and sending them to the appropriate destination. Every processor communicates with every other and the whole system keeps in lock-step.
- HYPERCUBE runs on 2^d processors and is a parallel version of a divide and conquer algorithm for transporting a matrix (see[9]). Each processor communicates with d other processors where each set of communications involves sending half of the matrix stored on that processor and receiving half of the matrix currently stored on a neighbour. After the communications phase, sub-matrices are transposed independently on each processor.

The HYPERCUBE benchmark was taken from a demonstration code written for the Intel iPSC/2 and it is informative to examine the dramatic improvements in performance that were possible by careful optimisation.

The initial version of this benchmark was a straight translation of the code written for the Intel iPSC/2-VX, which has relatively slow vector processors on each node. The particular code was written to demonstrate the non-numerical use of the Intel vector processors and the translation's performance on SUPRENUM was poor: a 1024 X 1024 transpose on 16 nodes required approximately 3.2 seconds.

The initial code used small buffers (ie. a column at a time) and a send / receive arrangement (ie. one node sends while the other receives). By changing that to an exchange arrangement (both send, then both receive), the time went down to 2.4 seconds. By using big buffers (half of the matrix on the node), the time was 1.8 seconds. All of these used asynchronous communication.

With synchronous communications, the time for the same problem using small buffers was 2.1 seconds. When large buffers were used, the time dropped to a dramatic 1.1 seconds. A detailed analysis of the different components of the overall time showed that considerable portion of time was taken up in the indirect addressing used to move data to and from the buffers. When this was rewritten using direct addressing the time dropped yet again, to about 0.71 seconds. By exploiting full compiler optimisation on the set-up loops, the time dropped again, to 0.42 seconds. Thus, performance was improved by over a factor of 7.5 without too much difficulty, but with some careful analysis.

The decrease in the number of communications caused by using larger buffers was significant even when just two nodes were used. The time required to transpose a 32 by 32 matrix with small buffers and an exchange arrangement was 78 ms. Small buffers and synchronised communication required the same time. If large buffers were used, with the exchange communications the time dropped to 27 ms, but if synchronous communications were used, the time was only 17 ms. Using direct, rather than indirect addressing decreased this to 8 ms. This corresponds to nearly a ten-fold performance improvement.

It appears that the major optimisation gain was through large buffers, which cut the communications overhead considerably. As an additional benefit, the communications set-up took place within one block of loops, which the compiler could optimise, leading to another performance gain.

The main benefit of synchronous communications over asynchronous was the reduction in the vector copies that were required by the system, it also reduced the overall memory requirement significantly when large buffers were used, allowing larger problems to be tried.

The ultimate performance is encouraging and has implications for key application components such as a parallel radix 2 2-D FFT. Here a series of 1-D FFTs are performed independently across the processors, the data is transposed and then another series of independent 1-D FFTs are performed. Using the previously quoted multiple single node FFT and this transpose, we predict that a 1024 by 1024 2-D FFT could be performed in about 2.6 seconds, which corresponds to a floating point rate of over 40 Mflop/s. An identical technique could be applied to radix 4 1-D FFTs, so that a $4^{10} = 1048576$ long FFT could also be transformed in 2.6 seconds with a floating point rate of 40 Mflop/s.

7. Conclusions

The benchmark programs from the Genesis pre-study provided a reasonable starting point for much of the SUPRENUM evaluation, but in order to obtain a good overview of the system, several new benchmarks, such as the transpose and general relatively benchmarks, were added. In addition, extensive changes were made to several other benchmarks, such as the FFT routines.

By concentrating on one main architecture - SUPRENUM, groups cooperated to overcome common difficulties and there was a beneficial synergy established. This might not have been as evident if several different platforms had been given roughly equal importance. The need to write code in a particular style to aid optimisation, along with the encouraged use of Fortran-90 array constructs furthered this synergy by providing de facto programming standards. This also complemented the previously mentioned effort to develop standards related to methodology and presentation.

In the rush to obtain the highest possible performance on SUPRENUM, some of these standards were sacrificed. For instance, not every benchmark had a corresponding Fortran-77 single node baseline code written for it. Sometimes the sequential code was lacking, other times it involved extensive use of Fortran-90 extensions. One of the major activities related to benchmarking in the 1991 work has involved filling in many of these holes.

The evaluation exercise and the related work to develop a portable benchmark has taught us a great deal about which programming standards to concentrate upon. For example, many benchmarks used similar communication operations, such as broadcast or global reduction operations. There was considerable variability in the way in

which these were written, with a matching variability in performance. Similarly, many benchmarks could have been improved, and optimised more quickly, if a standard and optimised single-node library had been available. The basic communication routines have been standardised in 1991, using the PARMAC macros [10] developed at GMD and Argonne National Laboratories. Work will continue on including higher level communication operations, written in terms of the PARMACs, in the benchmark suite. The level 1, 2 and 3 BLAS are part of the benchmark and optimised versions of these form the cornerstone of any standard single node library. Identifying and moving common single node computational operations to a standard base library will be an on going task for sometime as the benchmark evolves.

Evaluation of any computer, particularly a DMMP computer, is hard and intensive work. It is our hope that the benchmarks that have evolved from the SUPRENUM evaluation and Genesis pre-study will make this task considerably easier.

References

- [1] A. J. G. Hey, The Genesis Benchmarks, report June 1990, submitted to Parallel Computing (1991).
- [2] M. Metcalf and J. Reid, Fortran-90 Explained, Oxford Science Publications/OUP, Oxford and New York, (1990) Chapter 6.
- [3] M. Berry, D. Chen, P. Koss, D. Kuck, S. Lo, Y. Pang, L. Pointer, R. Roloff, A. Sameh, E. Clementi, S. Chin, D. Schneider, G. Fox, P. Messina, D. Walker, C. Hsiung, J. Schwarzmeier, K. Lue, S. Orszag, F. Seidl, O. Johnson, R. Goodrum, J. Martin, The PERFECT club benchmarks: effective performance evaluation of supercomputers, Intl. J. Supercomputer Appls. 3(3) (1989) 5-40.
- [4] SPEC Benchmarks Suite Release 1.0, SPEC Newslett. 2(3) (1990) 3-4. publ. Systems Performance Evaluation Cooperative, Waterside Associates, Fremont, California.
- [5] A. Friedli, W. Gentzsch, R. Hockney and A. van der Steen, A European Supercomputer Benchmarks Effort, Supercomputer 34, VI-6 (1989) 14-17.
- [6] P. Messina, C. Baillie, E. Felten, P. Hipes, R. Williams, A. Alagar, A. Kamrath, R. Leary, W. Pfeiffer, J. Rogers, D. Walker, Concurrency: Practice and Experience, 2(3) (1990) 195-255.
- [7] Proceedings 2nd International SUPRENUM Colloquium, 30 Sept.-2 Oct. 1987, Bonn, FRG. Special Issue (U. Trottenberg ed.), Parallel Computing, 7 (3) (1988)
- [8] R.W. Hockney and C.R. Jesshope, Parallel Computers 2: Architecture, Programming and Algorithms, Adam Hilger/IOP Publishing, Bristol & New York, (1988).
- [9] O.A. Mc Bryan and E.F. Van der Velde, Hypercube Algorithms and Implementations, SIAM Sci. Stats. Comp, 8(1987), pp. s227-s287.
- [10] R. Hempel, The Argonne / GMD Macros in FORTRAN for Portable Parallel Programming using the Message Passing Programming Model, Genesis Working paper, April 1991.

CASTLE: A TOOL FOR BAYESIAN LEARNING

S. ACID, L.M. DE CAMPOS, A. GONZALEZ,

R. MOLINA, N. PEREZ DE LA BLANCA

Departamento de Ciencias de la Computación e I.A.

Universidad de Granada. 18071 Granada, España.

SUMMARY

Causal networks codify relevance by means of proximity relationship in a graph and permit us to modify our knowledge by making only local consultations. The problems of learning such networks from direct empirical examples are considered here. The learning algorithms we have so far implemented in CASTLE, (Causal Structures From Inductive Learning) are described. Finally, we use CASTLE in a simple digit recognition problem.

1 INTRODUCTION

The aim of this paper is to describe the learning causal network algorithms and software currently being developed by members of the Department of Computer Science and Artificial Intelligence at the University of Granada for the ESPRIT project *Statlog* ([15]).

These algorithms are part of CASTLE, a software which so far allows the user to learn polytree's structure from raw data, propagate knowledge through out a polytree either interactively or in batch mode, simulate data from a given causal network and create and edit causal networks.

This paper is divided in the following sections. In section 2, we describe the concept of causal network and how they can be used to represent knowledge about a problem. Section 3 is a brief description of what learning is about in the context of causal networks. Section 4 describes the learning algorithms implemented in CASTLE to learn the skeleton of polytrees. Section 5 is devoted to the task of recovering the directions of the branches of a polytree-dependent distribution. Sections 6 describes other important parts of CASTLE. Section 7 is devoted to the application of the Bayesian learning methodology to a digit recognition problem. Finally, in section 8 we describe our future line of research.

2 CAUSAL NETWORKS

It is obvious that interpreting the rules in reasoning systems as conditional probability sentences like $P(B|A) = p$ will not allow any computation, unless it can be assured that the knowledge base only contains the proposition A . When a new fact, K , appears in the knowledge base, we should use $P(B|A, K)$ instead of $P(B|A)$. Probabilistic sentences would be inoperant if we could not verify that anything else present in the knowledge base is irrelevant for the calculation of the certainty of B .

The basic idea of causal networks is therefore to codify the knowledge in such a way that the information we cannot ignore for a particular reasoning task can be quickly identified and easily accessible.

In probability theory, relevance is identified with dependence. Obviously, dependence can be defined in terms of probability and conditioning. Two propositions A and B are said irrelevant (or independent) in a context C if

$$P(A|B, C) = P(A|C) \quad (1)$$

Causal networks codify relevance by means of proximity relationships in a graph, and permit us to modify our knowledge by making only local consultations. The direct dependence relationships can be stated directly and explicitly, preserving them as a stable part of the model.

Let us now make the ideas more formal.

As defined by Pearl, ([8]): Causal networks are directed acyclic graphs (DAGs) in which the nodes represent propositions (or variables), the arcs signify the existence of direct causal dependencies between the linked propositions, and the strengths of these dependencies are quantified by conditional probabilities.

The structure of a causal network can be determined in the following way (see figure 1): Each variable in the domain is identified with a node in the graph. We then draw arrows to each node X_i from a set of nodes $C(X_i)$ considered as direct causes of X_i .

The strengths of these direct influences are quantified by assigning to each variable X_i a matrix $P(X_i|C(X_i))$ of conditional probabilities of the events $X_i = x_i$ given any combination of values of the parent set $C(X_i)$. The conjunction of these local probabilities defines a consistent global model, i.e., a joint probability distribution. For example, for the causal network in figure 1, the joint distribution can be obtained as the product

$$P(x_1, x_2, x_3, x_4, x_5) = P(x_5|x_3, x_4)P(x_4|x_2)P(x_3|x_1)P(x_2|x_1)P(x_1) \quad (2)$$

Variables: X_1 to X_5

Influences:

X_1 is a direct cause of X_2

X_1 is a direct cause of X_3

X_2 is a direct cause of X_4

X_3 and X_4 are direct causes of X_5

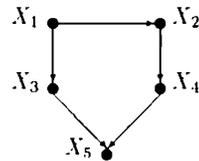


Figure 1: A Causal Network

Let us now look at the way a DAG represents the (in)dependences involving two or three variables A , B and C :

1. *Marginal and Conditional Independence:*

The two variables A and B have not any mutual influence: the knowledge about the value of the variable A never affects or modifies our belief about the value of variable B and vice versa. In a graph, these two variables are not connected by any path (they belong to two different connected components).

2. *Marginal and Conditional Dependence:*

The variable A is one of the causes of B or vice versa. The knowledge about the

value of one of the two variables always has influence on the value of the other variable. This relation may be represented by means of one of the two following subgraphs: $A \rightarrow B$ or $B \rightarrow A$.

There are more complex dependence/independence patterns involving three variables that are of a dynamic nature:

3. *Marginal Independence and Conditional Dependence:*

Two variables A and B are not dependent between each other merely by having influence on a common consequence C (A and B are marginally independent) but become dependent when we know the true value of C (A and B are conditionally dependent given C). This pattern can be represented by the following subgraph:

$$A \rightarrow C \leftarrow B \quad (3)$$

As an example of this dynamic dependence/independence relationship we could consider the following variables in cooking:

- A = Quality of ingredients,
- B = Skill of the cook,
- C = Quality of the dish.

4. *Conditional Independence and Marginal Dependence:*

If we know the value of a variable C then our knowledge about the value of the variable A has no influence on our knowledge about the value of the variable B (A and B are conditionally independent given C), but when we do not know the value of C , any knowledge about A (resp. B) is relevant for B (resp. A) (A and B are marginally dependent). This pattern can be represented by means of any of the three following subgraphs:

$$A \leftarrow C \rightarrow B \quad (4)$$

$$A \rightarrow C \rightarrow B \quad (5)$$

$$A \leftarrow C \leftarrow B \quad (6)$$

An example of (4), based on illness, could be:

- A = Body temperature
- B = Coughing
- C = Kind of illness

Without knowing the illness, our knowledge about one symptom (coughing) will influence our opinion about the existence of another symptom (fever). Once we know the illness, these symptoms become independent.

As an example of (5) we could consider the following variables:

- A = It's raining (yes or no)
- B = Slipping over (yes or no)
- C = Ground state (wet or dry)

Without knowing C , A and B are dependent but once C , the direct cause of B , is known A and B become independent.

Remark: Patterns 3 and 4 can be stated more generally for three sets of variables instead of just three variables, but for the kind of networks we will study it suffices to consider only variables.

So, given the following DAGs involving three variables displayed in figure (2), (i), (ii) and (iii) are equivalent in the sense that the three of them represent the fact that A and B are independent given C , while (iv) represents the marginal independence of A and B , events which become dependent once C is known.

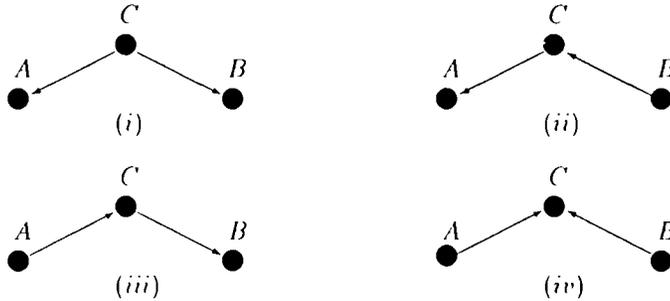


Figure 2: *Patterns of (in)dependency.*

Let us finally study the parameters required for the DAGs displayed in figure 2. (ii) will require $P(B)$, $P(C|B)$ and $P(A|C)$. Since

$$P(B)P(C|B) = P(B, C) = P(C)P(B|C) \quad (7)$$

(i) can be obtained from (ii), now changing A and C , (iii) can be obtained from (i). However, the model (iv) is quite different; its parameters are $P(A)$, $P(B)$ and $P(C|A, B)$ which cannot be determined from any of the previous sets.

It is important to note that the pattern of marginal independence and conditional dependence can be identified with only one graph structure, while the pattern of marginal dependence and conditional independence may be associated with three different causal graphs. Therefore, from dependence/independence relationships one can discriminate between two kinds of causal relationships (two causes with a common consequence and something else) but it is not possible to distinguish between a cause with two consequences and a cause with a consequence that in turn causes another consequence (if we consider the three variables isolated). Although limited, this capacity to identify causal relationships from dependence/independence ones constitutes the basis to build algorithms for learning causal structures from a set of examples.

3 BAYESIAN LEARNING

As we have said before, causal networks are tools capable of organising the knowledge to represent and manipulate relationships of relevance. Moreover, once the knowledge has been codified in a network, it facilitates a rapid response to inference tasks using a minimal amount of memory.

Once the network is constructed it constitutes an efficient device to perform probabilistic inferences. The problem of building such a network remains. The structure and conditional probabilities necessary for characterising the network could be provided either externally by experts or from direct empirical observations.

Under the Bayesian approach, the learning task in causal networks separates into two highly related subtasks, *structure learning*, that is to identify the topology of the network, and *parameter learning*, the numerical parameters (conditional probabilities) for a given network topology.

CASTLE so far is focused on learning structure rather than parameters, although obviously it also need to do some parameter estimation in order to produce a complete causal network.

Since a model with too many links is computationally useless, as it requires too much storage and lengthy procedures to produce predictions or explanations, it is essential that we give the learning process a built-in preference toward simple structures, those that have the fewest possible parameters and embody the fewest possible dependencies.

CASTLE focus on a particular kind of causal structures: polytrees (singly connected networks), networks where no more than one path exists between any two nodes. As a consequence, a polytree with n nodes has no more than $n - 1$ links. It is in polytrees (and specially in trees) where the ability of networks to decompose and modularise the knowledge attains its ultimate realisation. Polytrees does not contain loops, that is, undirected cycles in the underlying network (the network without the arrows or skeleton) and this fact allows a local extremely efficient propagation procedure (see [8]).

Let us now more formally define the concepts of polytree dependent distribution and nondegeneracy.

A given distribution $P(x)$ of n discrete value variables can be represented by some polytree F_0 if $P(x)$ has the form

$$P(x) = \prod_{i=1}^n P(x_i | x_{j_1(i)}, x_{j_2(i)}, \dots, x_{j_m(i)}) \quad (8)$$

where $\{x_{j_1(i)}, x_{j_2(i)}, \dots, x_{j_m(i)}\}$ is the (possibly empty) set of direct parents of the variable X_i in F_0 , and the parents of each variable are mutually independent, i.e.,

$$P(x_{j_1(i)}, x_{j_2(i)}, \dots, x_{j_m(i)}) = \prod_{k=1}^m P(x_{j_k(i)}) \quad (9)$$

Following ([8]) we say that a distribution $P(x)$ is nondegenerate if there exists a connected directed acyclic graph (DAG) that displays all the dependencies and independencies embedded in $P(x)$.

The process of learning a polytree structure is divided in two parts: learning the skeleton and directing it. First, let us see how to estimate the skeleton.

4 LEARNING POLYTREE'S SKELETON

In this section we will describe how CASTLE learns about the skeleton (the graph stripped of the arrows) of a nondegenerate distribution $P(x)$ that can be represented by a polytree.

The following theorem can be proven.[8], :

Theorem 1 *If a nondegenerate distribution $P(x)$ is representable by a polytree F_0 , then any Maximum Weight Spanning Tree (MWST) where the weight of the branch connecting X_i and X_j is defined by*

$$I(X, Y) = \sum_{x,y} P(x, y) \log \frac{P(x, y)}{P(x)P(y)} \quad (10)$$

will unambiguously recover the skeleton of F_0 .

Examining the proof of the above theorem it can be seen that we can use any D_{cp} function satisfying

$$\min(D_{cp}(X, Y), D_{cp}(Y, Z)) > D_{cp}(X, Z) \quad (11)$$

for any of the patterns $X \rightarrow Y \rightarrow Z$, $X \leftarrow Y \leftarrow Z$, $X \leftarrow Y \rightarrow Z$ and $X \rightarrow Y \leftarrow Z$ and not only the Kullback–Leibler measure defined in (10).

- In the learning skeleton menu in CASTLE we have implemented the following D_{cp} functions:

1. Information

- Kullback–Leibler

$$D_{cp}(X, Y) = \sum_{x,y} P(x, y) \log \frac{P(x, y)}{P(x)P(y)} \quad (12)$$

- Rajski, ([10]),

$$D_{cp}(X, Y) = - \frac{\sum_{x,y} P(x, y) \log \frac{P(x, y)}{P(x)P(y)}}{\sum_{x,y} P(x, y) \log P(x, y)} \quad (13)$$

2. L1–norm

- Unweighted

$$D_{cp}(X, Y) = \sum_{x,y} |P(x, y) - P(x)P(y)| \quad (14)$$

- Weighted

$$D_{cp}(X, Y) = \sum_{x,y} P(x, y) |P(x, y) - P(x)P(y)| \quad (15)$$

3. L2–norm

- Unweighted

$$D_{cp}(X, Y) = \sum_{x,y} (P(x, y) - P(x)P(y))^2 \quad (16)$$

- Weighted

$$D_{cp}(X, Y) = \sum_{x,y} P(x, y) (P(x, y) - P(x)P(y))^2 \quad (17)$$

4. L ∞

$$D_{cp}(X, Y) = \max_{x,y} |P(x, y) - P(x)P(y)| \quad (18)$$

Some of the D_{cp} functions listed above satisfy (11) (see [4] for details).

Although some of the functions listed above verifying (11) are faster to compute than the measure of Kullback–Leibler that is not the only reason to offer the user of CASTLE more than one function.

In our software we will never have $D_{cp}(X, Y)$ but an estimation $\widehat{D}_{cp}(X, Y)$ obtained from a file of examples. So we would like to use a function $D_{cp}(X, Y)$ such that the properties of $D_{cp}(X, Y)$ relatives to (11) are kept by $\widehat{D}_{cp}(X, Y)$. Research is currently being carried out to establish how robust our D_{cp} functions are.

Let us see how CASTLE learns a polytree structure. Let us assume we have a sample data file and we want to learn about the causal network representing the probability distribution where the data come from.

The first step to learn about the polytree associated to the read data sample is to estimate the skeleton of such polytree (see [8]).

Before learning the skeleton let us see how to introduce constraints on its topology.

CASTLE has an option allowing the user to include constraints on the skeleton topology, linking in advance those couples of nodes on which you have strong evidence on their dependency. According to the kind of evidence you have on the nodes you will be interested in introducing **Arrows**, **Edges** (branches) or boths. **Edges** means causal dependence between variables without specifying precedence between them. **Arrows** means causal dependence relationship, the sense of the arrow indicating the cause and effect node respectively.

Once you have finished introducing constraints, CASTLE will have stored the constraints together with the network structure, giving you the opportunity to estimate the skeleton with or without constraints.

CASTLE estimates the skeleton associated to a data set using different algorithms based on the Chow-Liu algorithm. This algorithm is based in a maximum weight spanning tree (MWST) algorithm. The method used to find the MWST is just Kruskal's algorithm, ([5]). Its performance is at most $O(n^2 \log(n))$ where n is the number of nodes. We could have used faster algorithms but the constrains are very easy to deal with when using Kruskal's algorithm.

5 DIRECTING THE SKELETON OF A POLYTREE

Having found the polytree's skeleton we move on to find the direction of the branches. To direct the branches we use the following facts: nondegeneracy implies that for any pair of variables (X_i, X_j) that do not have a common descendent we have

$$Dcp(X_i, X_j) > 0 \quad (19)$$

Furthermore, for the pattern

$$X_i \rightarrow X_k \leftarrow X_j \quad (20)$$

we have

$$Dcp(X_i, X_j) = 0 \text{ and } Dcp(X_i, X_j|X_k) > 0 \quad (21)$$

where Dcp is any of the functions defined in section 4 and the conditional Dcp functions $Dcp(X_i, X_j|X_k)$ are defined as the mean with respect to $P(x_k)$ of the functions $Dcp(X_i, X_j|X_k = x_k)$ which are analogous to $Dcp(X_i, X_j)$ but each marginal distribution being replaced by the corresponding conditional distribution given $X_k = x_k$, and for any of the patterns

$$X_i \leftarrow X_k \leftarrow X_j, \quad X_i \leftarrow X_k \rightarrow X_j \text{ and } X_i \rightarrow X_k \rightarrow X_j \quad (22)$$

we have

$$Dcp(X_i, X_j) > 0 \text{ and } Dcp(X_i, X_j|X_k) = 0 \quad (23)$$

Taking all these facts into account we can recover the head to head patterns, (20), which are the really important ones. The rest of the branches can be assigned any directions as long as we do not produce more head to head patterns.

So far so good but, what happens when we do not have the real distribution to calculate the Dcp function but a sample from it. We already have made some hypothesis

concerning \widehat{Dcp} to recover the skeleton of the polytree, however, now comes a greater problem. Conditions like

$$\widehat{Dcp}(X_i, X_j) = 0 \text{ when } X_i \text{ and } X_j \text{ are marginally independent} \quad (24)$$

or

$$\widehat{Dcp}(X_i, X_j|X_k) = 0 \text{ when } X_i \text{ and } X_j \text{ are conditionally independent given } X_k \quad (25)$$

are hardly satisfied. There may even be room for inconsistencies: we may have for the skeleton shown in figure 3 the following inequalities

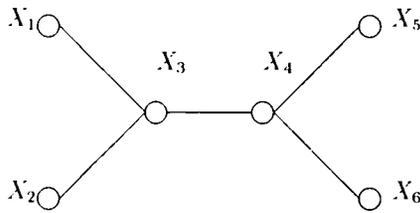


Figure 3: *Skeleton of the Polytree*

$$\begin{aligned} \widehat{Dcp}(X_1, X_2) &< \widehat{Dcp}(X_5, X_6) < \widehat{Dcp}(X_1, X_4) \\ &< \widehat{Dcp}(X_3, X_5) < c < \widehat{Dcp}(X_2, X_4) < \widehat{Dcp}(X_3, X_6) \end{aligned} \quad (26)$$

where c is a threshold used to detect independencies.

In the example displayed in figure 3 we can see that since X_1 and X_2 are independent then the arrows $X_1 \rightarrow X_3$ and $X_2 \rightarrow X_3$ should be in the polytree; as X_1 and X_4 are also independent, the arrow $X_3 \rightarrow X_4$ should also be in the polytree. But X_2 and X_4 are dependent; once the branch $X_2 - X_3$ points at X_3 , the branch $X_3 - X_4$ should point at X_4 instead of pointing at X_3 : there is then an inconsistency. The same thing happens with the branches $X_4 - X_5$, $X_4 - X_6$ and $X_3 - X_4$. From these two examples, it is obvious that some of the dependencies and/or independencies provided by the data should not be taken into account in order to obtain a polytree structure.

CASTLE in its current state implements two approaches to recover the orientation of the branches.

The first one works as follows on the recovered skeleton ([8]):

1. Search the internal nodes of the skeleton, beginning with the one having more neighbours left, until a multi parent node Y is found using one of the tests (27), (28) or (29) to find any possible structure of the form $X \rightarrow Y \leftarrow Z$ where X and Z denote adjacent nodes to Y in the skeleton.
2. Being Y multi parent, the nodes which are not parents of Y in the polytree become descendants of Y .
3. For each node C having one incoming arrow from Y , resolve the directionality of all of its remaining adjacent W branches using one of the test (27), (28) or (29) to find the structures of the form $Y \rightarrow C \leftarrow W$.
4. Repeat steps 1 through 3 until no further directionality can be discovered.

There may remain some branches without being assigned any direction. We will describe what to do later.

To find the causal structures having the form $X \rightarrow Y \leftarrow Z$, we can use

$$\widehat{Dcp}(X, Z) < c \tag{27}$$

where c is a constant chosen by the users,

$$\widehat{Dep}(X, Z) < \widehat{Dep}(X, Z|Y) \tag{28}$$

or a

$$\chi^2 - test\ of\ independence \tag{29}$$

on X and Z where the level of confidence is set to 0.95.

We will now describe the second approach to the problem of recovering the direction of the branches.

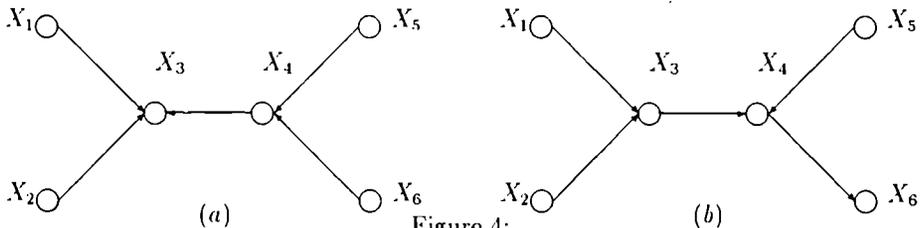
In the previous method for directing the skeleton, there is no criterion to decide which (in)dependencies should be preserved, and it does the selection blindly, depending on the order in which the nodes are examined.

This second method to direct the skeleton is based on the same criteria to distinguish head-to-head arrows, but provides some guidance to decide which (in)dependencies should be preserved in the case of conflict. The basic ideas are the following: (we will describe them for marginal dependency measures, the same apply to conditional dependency measures, see [4] for details)

- If $Dep(a,b) < Dep(e,d) < c$ (a and b , e and d are independent), then the independence between a and b has more priority and must be preserved first.
- If $c < Dep(a,b) < Dep(e,d)$ (a and b , e and d are dependent), then the dependence between e and d has more priority and must be preserved first.

Moreover, the user may decide whether he/she prefers to preserve dependencies first or independencies first, using always the above rules between dependencies or between independencies. This corresponds with an orientation method based on a search with priorities.

If we use the independence first criterion for the example shown in (26) for figure 3, we obtain figure 4(a). As we see, we have lost the dependence between X_2 and X_4 , and the independence of X_3 and X_5 . If we use the dependence first criterion, we obtain the polytree shown in figure 4(b). In this case we have lost the independencies of X_1 and X_4 and of X_5 and X_6 . In any case, we are losing the weaker (in)dependencies compatible with the selected criterion.



(a) polytree obtained with independency first. (b) polytree obtained with dependency first.

We can use another selection criterion. To transform the range of variation of both independencies, $[0, c]$, and dependencies, $[c, maximum\ value\ of\ Dcp(...)]$ into a common

range in which we can compare the strengths of them, thus preserving first the strongest (in)dependencies. All these options have been implemented in CASTLE as **Searching with Priorities**, there is an analogous interval for the conditional criterion (28)).

Any of the methods the user can choose to find the direction of the branches may leave some of them undirected([8]). CASTLE **completes** the network assigning direction to the undirected branches, constrained to not producing more head to head patterns. This is the last step in the CASTLE learning process. After using **Complete** you will not be able to go back in the learning process to change the topology in your network.

Finally, let us examine what happens when the distribution $P(x)$ cannot be represented by a polytree. The theory does not say anything about how close the obtained polytree dependent distribution $P^p(x)$ is to $P(x)$ (see section 8). Chow and Liu, ([8]), however, show how to obtain the distribution $P^t(x)$, tree-dependent distribution, having the form

$$P^t(x) = \prod_{i=1}^n P(x_i | x_{j(i)}) \quad (30)$$

where $X_{j(i)}$ is the variable designated as the parent of X_i in some orientation of the tree which is closer to the distribution $P(x)$ in the sense of minimising

$$D(P, P^t) = \sum_x P(x) \log \frac{P(x)}{P^t(x)} \quad (31)$$

As far as the skeleton is concerned, a tree dependent distribution may have the same as a polytree dependent distribution, the only difference being that when directing the tree will not have any head-to-head pattern. The way of using CASTLE to learn the best tree dependent distribution, $P^t(x)$, approximating $P(x)$ is to choose in Skeleton the Kullback-Leibler information measure, and then move to complete.

6 MORE ABOUT CASTLE

CASTLE has been created to test and evaluate Bayesian Learning algorithms. To help in the achievement of such a goal, CASTLE has some interesting utilities. The user can edit a polytree, i.e., to draw nodes, link them by arrows, give name to the nodes and cases, and define the conditional or marginal probabilities in each node. This option can be combined with a simulation process to offer a way of testing the performance of the algorithms implemented (see [2] for details).

We have recently included in CASTLE the possibility of propagate knowledge through out a polytree using the method described in [8]. Using this module the learned net can be consulted to reason about the interpretation of specific input data. The interpretation process involves instantiating a set of variables corresponding to the input data, calculating its impact on the probabilities of a set of variables designated as hypotheses, and, finally, selecting the most likely combination of these hypotheses.

There is now a batch version of CASTLE, the name of the program is **xbcastle**. This version allows the user to execute the learning algorithms in batch mode, in this case, **xbcastle** creates a file containing the estimated network. Furthermore, the user can provide **xbcastle** with a new type of file containing a set of samples of observed values of any variable but the last one (the one thought as classifier), in this case, **xbcastle** propagates the observed knowledge through out the net and outputs a file containing the posterior probability of the cases of the classifier given the observed values of the rest of the variables (see [2] for details).

Digit = X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7
0	1	1	1	0	1	1	1
1	0	0	1	0	0	1	0
2	1	0	1	1	1	0	1
3	1	0	1	1	0	1	1
4	0	1	1	1	0	1	0
5	1	1	0	1	0	1	1
6	1	1	0	1	1	1	1
7	1	0	1	0	0	1	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	0

Table 1:

7 EXAMPLE

Let us now illustrate the use of the Bayesian learning methodology in a simple model, the *digit recognition in calculator*.

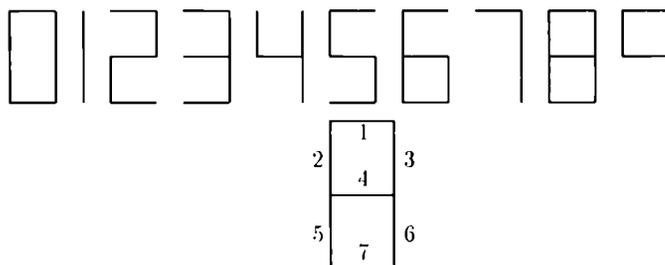


Figure 5:

Digits are ordinarily displayed on electronic watches and calculators using seven horizontal and vertical lights in on/off configurations (see figure 5). Let us number the lights as shown in figure 5.

Let us take $X = (X_0, X_1, X_2, \dots, X_7)$ be an eight-dimensional vector where $X_0 = i$ denote the i th digit, $i = 0, 1, 2, \dots, 9$ and when fixing X_0 to i the remaining vector (X_1, X_2, \dots, X_7) is a seven-dimensional vector of zeros and ones with $x_m = 1$ if the light in the m position is on for the i th digit and $x_m = 0$ otherwise. Our aim is to build up the polytree displaying the (in)dependencies in X .

The values taken by X are shown in table 1.

Let us use these ten eight-dimensional vectors as learning sample to generate the *Maximum Weight Spanning Tree*, that is the skeleton of the polytree, where the weight of the arcs is defined by the Kullback-Leibler information measure.

The recovered skeleton is shown in figure 6. We now direct the skeleton using the *Marginal Information* criterion. CASTLE then tells us what we had expected:

X_i and X_j are conditionally independent given X_0 , $i, j = 1, 2, \dots, 7$

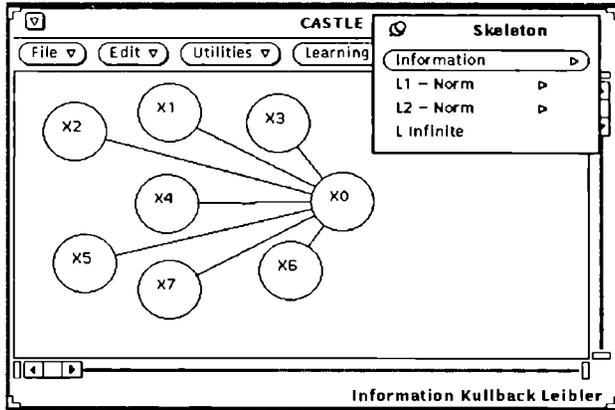


Figure 6: *Finding the Skeleton*

Let us briefly discuss the obtained polytree. The model correctly classify the “perfect” digits. For instance we have

$$P(X_0 = i | X_1 = 0, X_2 = 0, X_3 = 1, X_4 = 0, X_5 = 0, X_6 = 1, X_7 = 0) = \begin{cases} 1 & \text{if } i = 1 \\ 0 & \text{if } i \neq 1 \end{cases}$$

However, it has not predictive power, since modifying the light of just one vertical or horizontal line rules out any digit that does not satisfy the condition, for instance

$$P(X_0 = i | X_1 = 0) = \begin{cases} 1/2 & \text{if } i = 1, 4 \\ 0 & \text{otherwise} \end{cases}$$

Let us now study how to improve the predictive power of the model. To do so, let us generate examples from a faulty calculator.

Each of the seven lights has probability 0.1 of not doing what is supposed to do. More precisely, the data consist of outcomes from the random vector $X_0, Y_1, Y_2, \dots, Y_7$ where X_0 is the class label, the digit, and assumes the values in $0, 1, 2, \dots, 9$ with equal probability and the Y_1, Y_2, \dots, Y_7 are zero-one variables. Given the value of X_0 , the Y_1, Y_2, \dots, Y_7 are each independently equal to the value corresponding to the X_i given in table 1 with probability 0.9 and are in error with probability 0.1.

For example

$$P(Y_1 = 0, Y_2 = 0, Y_3 = 1, Y_4 = 0, Y_5 = 0, Y_6 = 1, Y_7 = 0 | X_0 = 1) = (0.9)^7$$

and

$$P(Y_1 = 0, Y_2 = 0, Y_3 = 1, Y_4 = 0, Y_5 = 0, Y_6 = 1, Y_7 = 0, X_0 = 1) = (0.9)^7 \times \frac{1}{10}$$

Let us generate two hundred samples of this distribution and use them as learning sample. After reading in the sample, estimating the skeleton using *Kullback Leibler*

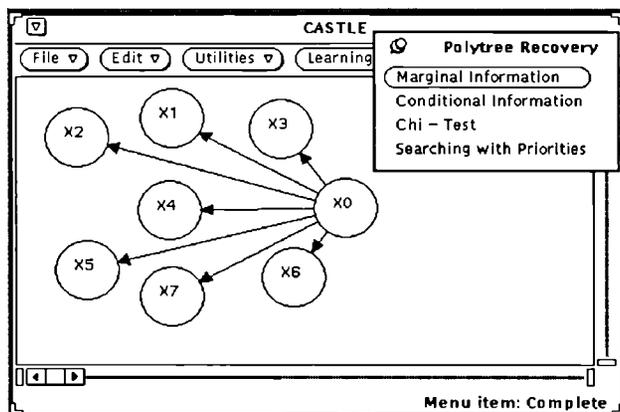


Figure 7: Estimated Polytree

Digit	0	1	2	3	4	5	6	7	8	9
0	463	0	2	0	0	0	519	0	16	0
1	0	749	0	0	0	0	0	251	0	0
2	1	0	971	0	6	0	1	12	0	0
3	1	0	0	280	0	699	19	2	0	0
4	0	21	0	0	913	0	0	1	2	63
5	290	0	0	0	0	614	51	5	10	0
6	0	5	0	0	432	0	0	0	563	0
7	6	0	0	0	0	78	7	810	0	99
8	26	0	230	693	1	0	46	4	0	0
9	125	0	0	0	0	3	0	359	43	470

Table 2: Probability $\times 1000$ for some "digits"

Information, directing the skeleton using *Marginal Information* and finally completing it, the polytree estimated by CASTLE is the one shown in figure 7.

Let us now examine the predictive power of this polytree. The posterior probabilities of each digit given some observed patterns are shown in table 2.

We now see that the digit need not be "perfect" to be identified with some degree of accuracy.

Before finishing this section we would like to note on two points. The first refers to the obtained polytree. The reader has probably observed that when using the vector $(X_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7)$ we should have used a network with seven hidden nodes. However, we have preferred to keep the model simply here. The second refers to the chosen example. The election of the example is not arbitrary. The same problem has been used to illustrate the use of Boltzmann Machines ([1]) and CART (Classification and Regression Trees) Techniques([6]), see [11] for very interesting comments on the problem. Our aim has been to prove that although simple the Bayesian learning techniques are

powerful tools.

8 WORK IN PROGRESS

Although as it is at the moment CASTLE is a limited tool for learning structure from raw data, we intend to research on and include the following possibilities in the software:

- The use of more complex searching techniques to minimise the cost, measured as a function of the inconsistencies, of the recovered polytree. The **searching with priorities** option is just a first approach to that job. However, we intend to use uniform cost search, best first search and A algorithms based on some sort of heuristics. We also intend to use combinatorial optimization techniques, ([1]), to recover the polytree's structure.
- Study how close the recovered polytree-dependent distribution is to the real distribution when this cannot be represented by a polytree.
- Amplify the **set constraints** option to allow the user to include other kind of restrictions like: keeping some variables independent, forcing some variables to be independent when others appear dependent and so on.

These are the first steps we intend to take to improve CASTLE. We would also like to deal with the following problems in the future. Learning more complex causal structures ([9],[16]) and estimation of the parameters needed in the network ([12],[13]). To carry out this second task we could benefit from the Bayesian approach to Image Analysis ([11],[7]).

ACKNOWLEDGEMENT

CASTLE's graphical edition facilities are inspired on those developed in ENTORNO, a software written by the team of the ESPRIT project DRUMS at the DECSAI. The propagation software has also been borrowed from ENTORNO.

REFERENCES

- (1) AARTS, E. and KORST, J. (1990). *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*. John Wiley.
- (2) ACID, S., CAMPOS, L.M.de, GONZALEZ, A., MOLINA, R. and PEREZ de la BLANCA, N. (1991) CASTLE: Causal Structures from Inductive Learning. Release 2.0. *Technical Report no 91-4-3*. Dept of Computer Science and A.I. (DECSAI). University of Granada.
- (3) ACID, S., CAMPOS, L.M. de, GONZALEZ, A., MOLINA, R., PEREZ de la BLANCA, N. (1991), *Bayesian Learning Algorithms in CASTLE, Technical Report 91-4-2*, Department of Computer Science and Artificial Intelligence (DECSAI), University of Granada.
- (4) ACID, S., CAMPOS, L.M.de, GONZALEZ A., MOLINA, R. and PEREZ de la BLANCA, N. (1991) Some results on the automatic construction of Bayesian Networks. *Proc of the 16 S.O.R.* Trier, September, 1991.
- (5) AIHO, A.V., HOPCROFT, J.E. and ULLMAN, J.E. (1987) *Data Structures and Algorithms*. Addison-Wesley.

- (6) BREIMAN, L., FRIEDMAN, J.II., OLSHEN, R.A. and STONE, C.J. (1984). *Classification and Regression Trees*. Wadsworth.
- (7) MOLINA, R. and RIPLEY, B.D. (1989). Using spatial models as priors in image analysis. *J. Appl. Statist.*, **16**, 193-206.
- (8) PEARL, J. (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan and Kaufmann.
- (9) PEARL, J. and VERMA, T.S. (1990) A Formal Theory of Inductive Causation. *Technical Report R-155* Department of Computer Science. University of California.
- (10) RAJSKI, C. (1964) On the normed information rate of discrete random variables. *Trasl of the Thrid Praga Congress*, 583-585.
- (11) RIPLEY, B.D. (1988) *Statistical Inference for Spatial Processes*. Cambridge Univ. Press.
- (12) SPIEGELHALTER, D. J. (1986) Probabilistic reasoning in predictive expert systems. In *Uncertainty in Artificial Intelligence* (eds L.K. Kanal and J. Lemmer) 48-68. Amsterdam: North-Holland
- (13) SPIEGELHALTER, D. J. and LAURITZEN, S.L. (1990) Sequential Updating of Conditional Probabilities on Directed Graphical Structures. *Networks*, **20**, 579-605.
- (14) SPIRITES, P., GLYMOUR, C. and SCHEINES, R. (1991) *Casuality, Statistics and Search*. Unpublished manuscript.
- (15) STATLOG (1990). Technical Annex of ESPRIT project : *Comparative Testing of Statistical and Logical Learning*. *Statlog*.
- (16) VERMA, T.S. and PEARL, J. (1990) Equivalence and Synthesis of Causal Models In *Proc. Sixth Conference on Uncertainty in Artificial Intelligence*. Cambridge Mass, 220-227.

**MACRO AND MICRO FEATURES
FOR AUTOMATED PRONUNCIATION IMPROVEMENT
IN THE SPELL SYSTEM**

Steven M. Hiller, Edmund Rooney and John Laver
The Centre for Speech Technology Research
University of Edinburgh
80 South Bridge
Edinburgh EH1 1HN
Scotland

Maria-Gabriella Di Benedetto
University of Rome
"La Sapienza"
INFOCOM
Via Eudossiana, 18
00184 Rome
Italy

Jean-Paul Lefèvre
OROS S.A.
13, Chemin des Prés
ZIRST - BP 26
38241 Meylan cédex
France

SUMMARY

In this paper, the analysis of macro (prosodic) and micro (segmental) features is described for a workstation designed to improve the pronunciation of English, French and Italian by non-native speakers. The SPELL workstation is intended to be a teaching device aimed at intermediate ability foreign language learners. Audio and visual aids will be used to help students improve their general intelligibility within a basic teaching paradigm called **DELTA** (Demonstrate, Evaluate Listening, Teach and Assess). Prosodic analysis will apply to the features of intonation, stress and rhythm. A phonological approach is used for intonation which provides a well-structured system of contrasting units that correlate with discrete linguistic functions. A more limited approach to the prosodic phonology of stress and rhythm will be taught in the SPELL system by manipulating the relatively simple acoustic features of vowel quality and segmental duration. The micro feature analysis will focus on the segmental class of vowels. A distinctive feature approach is used to characterize non-native vowel pronunciation. Acoustic properties are sought which will be speaker-independent.

I. INTRODUCTION

SPELL (Interactive System for Spoken European Language Training) is a two year ESPRIT project which began in September 1990. Its main aim is the development of tools to be used in the automated assessment and improvement of non-native language pronunciation. This is a feasibility study involving English, French and Italian which will lead to an initial demonstrator system. The technical objectives of the project are to develop methods for analyzing the characteristics of speech produced by non-native speakers, to develop metrics for identifying differences between a non-native speaker's pronunciation and a

model offered by the system, and to provide user friendly feedback which will help to improve pronunciation.

The research and development of the SPELL workstation addresses many of the concerns of the ESPRIT program. Upon successful completion, the project will have produced a user friendly workstation which can be set alongside other learning tools in university and school language laboratories. The main technical innovation behind SPELL is the departure from the traditional practice of whole utterance matching used when teaching pronunciation. Instead, well-founded phonetic and phonological principles will be applied to teaching selected aspects of English, French and Italian pronunciation. The analysis of these three languages naturally requires a high degree of international co-operation within the consortium. Expertise in phonetics, signal processing and systems development is distributed across the project members. The partners in this collaborative project and their roles are as follows:

1. OROS S.A., 13 Chemin des Prés, ZIRST – BP 26, 38241 Meylan cédex, France (prime contractor, signal processing)
2. The Centre for Speech Technology Research, University of Edinburgh, 80 South Bridge, Edinburgh EH1 1HN, Scotland (phonetic analysis)
3. ALCATEL FACE Standard, Research Center, Via Nicaragua 10, 00040 Pomezia, Italy (signal processing)
4. Tecnopolis CSATA Novus Ortus, 70010 Valenzano, Bari, Italy (user interface)
5. University of Rome, "La Sapienza", INFOCOM, Via Eudossiana 18, 00184 Rome, Italy (phonetic analysis)

This paper begins with a general description of the framework of the SPELL system within which the proposed macro and micro feature analysis (as defined below) will operate. This general discussion sets out some basic assumptions about the system and the phonetic issues which it must address. A general teaching paradigm for foreign language pronunciation called DELTA is also proposed. The discussion then focuses on the analysis of macro and micro features to be used within the SPELL system. The *macro features* comprise the prosodic parameters of intonation, stress and rhythm. A unifying analytical approach for intonation is developed for the three target languages, making use of the concepts of *pitch anchor points* and *pitch trajectories*. A more limited approach to the prosodic phonology of stress and rhythm will be taught in the SPELL system by manipulating the acoustic features of vowel quality and segmental duration. The *micro feature* analysis will focus on the segmental class of vowels using a distinctive feature approach to characterize non-native vowel pronunciation. Acoustic properties are sought which will provide speaker independence for vowels.

2. A GENERAL DESCRIPTION OF THE SPELL SYSTEM

This section discusses some general issues in the development of the SPELL workstation, beginning with an outline of the basic assumptions on which the research will be based.

Some Basic Assumptions about the Spell System

Assumption 1. The SPELL workstation will be used as an autonomous teaching system.

A system for teaching foreign language pronunciation can be designed in two ways: 1) as an assistant workstation to a language teacher who organizes the particular pronunciation tasks to be practised by the student while using the system and 2) as an autonomous teaching system which is used without the need of direction by a teacher. The second approach is assumed here, thereby forcing a complete consideration of all the possible issues which might affect the performance of the system. Most importantly, courseware in some narrowly-defined topic areas will need to be created in order to achieve an autonomous teaching system.

Assumption 2. Users will be intermediate ability foreign language speakers.

This assumption avoids the need for the SPELL system to teach the basics of language use as well as pronunciation. It will need to be determined what is meant by "intermediate ability" and pronunciation tests will be devised accordingly for the three SPELL languages.

Assumption 3. Audio and visual aids will be available to the user.

These are two of the major technical interests of the SPELL project. The audio aids will enable the user to listen to the pronunciation of items of interest and will synthesize intermediate or exaggerated targets to attract the student's performance into the required zone of acceptability. Visual aids must produce intelligent displays which help a student to visualize relevant linguistic/phonetic concepts without requiring expert knowledge.

Assumption 4. Speaker intelligibility will be used to gauge pronunciation improvement.

Mimicry is often used by foreign language teachers in an attempt to achieve fluency in a given foreign language. In fact, there are very few foreign language learners who genuinely need to produce fully native versions of pronunciation. For the majority of students, improvement in intelligibility is a more practical objective in the successful use of a foreign language. It should be noted that mimicry is required to achieve success at pronouncing unfamiliar sounds but that functional ability to communicate with foreign listeners is of more interest than "speaking like a native" (see, for example, 1-3).

The Development of SPELL Courseware

In this section, some practical aspects in the development of courseware for the SPELL workstation are presented.

Courseware useful for improvements in pronunciation can be divided into two general types: *practice* courseware and *teaching* courseware. In the case of practice courseware, only quantitative feedback is provided to the student without any further directions from the system. The main drawback to this approach is that improvements in pronunciation performance are not very predictable. The use of teaching courseware provides directed instruction to the student, allowing better predictions to be made about the student's performance, and enabling a proper evaluation of the system itself as a teaching aid. The development of teaching courseware will concentrate research efforts towards a system which touches on all levels of pronunciation teaching.

The typical aspects common to many foreign language teaching practices can be described in a paradigm called DELTA. This system is described as follows:

Demonstrate — Audio demonstrations by the system of various utterances are used to highlight the pronunciation features of interest. For example, differences in vowel quality would be demonstrated by playing out words which contain minimal pairs such as *beet* versus *bit* for the /I/ versus /ɪ/ vowel distinction in English. The same pair of words might appear in sentences to demonstrate the phonological significance of the vowel distinction (e.g. "I would like a bit/beet.").

Evaluate Listening — Small listening tests are completed by the student to evaluate his or her ability to perceive the pronunciation features of interest. For example, the student would take an ABX test: the two utterances A) "bit" and B) "beet" are played to the student, who must decide if the test utterance X is the same as A or B. If the student fails to perceive the distinction for a given pronunciation feature then he or she may be asked to go back to the Demonstrate stage for more examples.

Teach — The actual teaching of the pronunciation features of interest takes place at this stage, with quantitative feedback for the student and directions for modifying inadequate performances.

Assess — This stage is the formal evaluation of the student's ability to pronounce the features of interest. For example, the student is given N attempts to produce utterances containing the features of interest. If the student achieves a certain percentage of correct pronunciations then he or she should proceed onto the next features to be taught; otherwise the student should go back to the Demonstrate phase for this particular task.

In addition to the DELTA paradigm, a fuller assessment of proficiency can be given to the students after several SPELL lessons have been completed in order to evaluate the student's general performance while using the workstation. For example, a cloze test could be given in which the features of interest are embedded within a section of contrived text. The text is designed in such a way that listening judges can evaluate the features of interest without the student being aware of which features are being tested. (Bem-

stein, Cohen, Murveit, Rtischev and Weintraub (4) studied the feasibility of automatically grading the performance of Japanese students speaking English and found a good correlation between evaluations completed by human judges and the automatic system.) Most importantly, this assessment program provides the SPELL project with a means of evaluating the performance of the demonstrator system.

Problem Areas for Research and Development of SPELL

A number of problem areas for the development of the SPELL workstation are discussed in this section.

The Integration Problem

A useful pronunciation-teaching system will not be created if it relies solely on the teaching of phonemes in isolation. In the early stages of teaching, isolated phonemes may be useful for simple demonstrations and training but more natural data must also be used in which phonemes have been integrated in real speech items or phrases.

The Equivalence Problem

In speech technology research, there is often an assumption of a direct equivalence between information contained in the acoustic speech waveform and the resultant phonetic perception of that signal. However, this belief in equivalence is not completely valid. For example, the ability to mimic a given pitch contour exactly does not guarantee any generalization of pitch use within a language. The reasons for this are twofold. Firstly, native listeners are very sensitive to the linguistic relevance of small alignment/adjustment details of a pitch contour, and the linguistic relevance of the contour therefore springs partly from its integration with the segmental performance. Secondly, a given pitch contour acquires its functional value partly from its relative placement in the pitch range of the speaker concerned, not from its absolute value. It is therefore more desirable to concentrate on getting the student to imitate more abstract aspects of the contour such as the location of the pitch peak and the shape of the contour. In general, the system needs to be able to judge when the student has produced an acceptable version of a given feature of interest in relative terms rather than by absolute matching techniques.

The Segmentation Problem

Accurate feature analysis will depend on the location of phonetic segments within an utterance produced by a student. The analysis of vowel quality (micro) features certainly requires the prior location of the vowel segments. Proper prosodic analysis too, cannot be completed without first locating those phonetic segments which have had durational and intonational features overlaid on their structures. Therefore, an automatic SPELL segmentation program is being developed for application to the speech signal prior to feature extraction and analysis.

Pronunciation Errors

The SPELL system must be able to deal with a number of different types of pronunciation error produced by non-native speakers of a given language. Firstly, there are the *structural* errors, brought from the mother tongue, which can take the form of additions to or omissions from the expected segmental sequence. For example, an Italian speaking English may tend to add vowels in order to preserve the syllabic structure of Italian, as in the word "bead" being pronounced /b i d ə/. Secondly, *systemic* errors can occur in which the phoneme of interest does not exist within the speaker's mother tongue and the closest native phoneme is used as a regular substitution (e.g. a French person speaking English might tend to pronounce the word "bit" as "beat"). Thirdly, there are *realization* errors in which a version of the non-native phoneme of interest does exist within the native speaker's system but it is still not quite pronounced correctly (e.g. a French person speaking English may pronounce the vowel in the word "bead" using the correct quality but with the wrong duration). Finally, the speaker may produce gross mistakes such as misreading, stuttering, false starts, etc.

How can the input waveform be segmented correctly when such errors may exist? Firstly, the test utterances will be constructed in such a way as to limit the types of error which might occur. Secondly, the SPELL segmenter has been designed using a segmental transition network which includes the more predictable types of error which occur between two given languages.

Feedback

The appropriate feedback will have to be provided by the SPELL workstation for the user. In the case of vowel quality, it is felt that quantitative feedback on its own (i.e. without diagnostic information) would be appropriate, since it may be difficult to convey complex articulatory relationships to the linguistically unsophisticated foreign language learner. For example, basic vowel quality distinctions can be demonstrated by displaying targets within a vowel diagram which the student must hit with a 'voice cursor' controlled by a vowel formant detection program. In effect, the student learns vowel quality distinctions via trial-and-error biofeedback rather than by active diagnostic feedback on the part of the SPELL system. Prosodic features appear to be more straightforward to describe to the student and therefore diagnostic feedback is appropriate (e.g. the workstation informs the student that he or she used a falling pitch contour at the end of the utterance but that a rising pitch would have been more appropriate). It should be emphasized that the feedback to a SPELL user will not be expressed in terms of explicit sophisticated linguistic or phonetic concepts: for example, the student need not be aware of the abstract phonological systems which form the bases of the training system.

3. THE ANALYSIS OF MACRO FEATURES IN THE SPELL SYSTEM

The preceding sections set out a framework for the analysis and remediation of non-native pronunciations of a given language. In this section, the prosodic aspects of pronunciation are examined in more detail.

Definition of Macro Features

Macro or *prosodic* features are those which operate over stretches of speech longer than the single segment or phoneme, and which may characterize an utterance as a whole. Features normally seen as prosodic include intonation, stress and rhythm.

Intonation is generally defined as the manipulation of pitch for linguistic and paralinguistic purposes at a level above that of the segment (e.g. 5: 83). All utterances, including words spoken in isolation, have an intonation "contour". **Stress** is the term used to refer to a number of ways in which certain syllables are made more prominent than surrounding syllables. Stress functions at two levels: within the word (lexical stress or accent) and within the utterance as a whole (rhythmic stress), where it is closely integrated with intonation and rhythm. The **rhythm** of an utterance is given by the patterning in time of the segments, syllables and stresses; its actual definition and description remain the subject of controversy, and its measurement in terms of acoustic features is notoriously difficult (6).

Phonological Approaches to Intonation

The analysis of speech at the segmental level into phonemes is taken for granted, and it is often not realized that there is a considerable body of linguistic theory underlying this analysis, which makes it possible to assume, for example, that the 'p' sounds produced in words such as "put", "top" and "spin" can be regarded phonologically as "the same thing" despite some major acoustic differences. The analysis of prosody is still not as developed as that of segmental phenomena, but it is recognized that there is a similar need for a theoretical basis to make sense of the multiplicity of acoustic realizations of prosodic features which are found when real speech is examined (7). This is particularly important for language teaching. For example, it is not possible to teach intonation simply by direct imitation of target utterances, since actual pitch contours can vary enormously; what the pupil requires is a pattern or model which can be generalized to other utterances of the same type or for the same purpose, and the ability to choose from a set of such models to convey contrasts of meaning or emphasis. This is the essence of phonological analysis:

a phonological treatment aims to establish a system of structures which can be used as a vehicle for meaning and a set of contrasting elements which can be inserted into those structures.

The development of a complete phonological description for each of the three target languages is beyond the scope of the SPELL project. Instead, published analyses of English, French and Italian will be used to achieve the aim of teaching the basic prosodic patterns of each language to students.

In intonation analysis, English has the most developed treatment with two main traditions: the so-called “British” school, which treats the pitch contour as the unit of analysis (e.g. 8–11); and the “American” school, which deals with distinctive pitch levels (e.g. 12–15). Rather less work has been done on French intonation (examples, in a variety of approaches, are 16–22). Analyses of Italian are fairly rare (23–25).

Comparing intonational systems amongst these three languages in terms of their phonology is quite difficult given the varying depth of treatment and the differing approaches to the problem. Some general principles are clearly common to all three languages. Firstly, pragmatic linguistic functions such as statements and questions are differentiated by opposing pitch movements (e.g. falling versus rising pitch). Secondly, pitch movements are related to rhythmical structure by the marking of accented syllables. Finally, intonational pitch movements are related to or *anchored* to the segmental structure of the utterance (this is the segmentation problem discussed above).

The major difference in terms of phonology between English, French and Italian is the extent to which the intonation contour is treated as a structural chain with elements of choice at certain locations. In French, the choices within the contour are very limited with the whole contour being treated as a single “tune” (see, for example, Leach (22)). Italian is slightly more complex in that the contour can be subdivided into a chain but with a limited choice of elements. In English, the contour can be subdivided into a very complex chain with many choices at various locations (see, for example, Halliday (11)). A practical approach to describing and teaching intonation has been adopted to overcome these differences in phonology between the three languages, as discussed below.

The Analysis of Intonation

This section presents the analysis of intonation for French, Italian and English for the SPELL workstation.

The phonetic transcription of pitch phenomena is conventionally achieved by limiting the transcription to comment on relative pitch movements within a speaker’s linguistic range. Two parallel horizontal lines are usually drawn to act as a staff representing the upper and lower limits of the linguistic *pitch span*; the use of horizontal lines means that the effect of pitch declination has been ignored (i.e. the tendency for fundamental frequency to slowly decrease from the beginning to the end of an utterance has been eliminated). This pitch transcription will be used in the remainder of the paper.

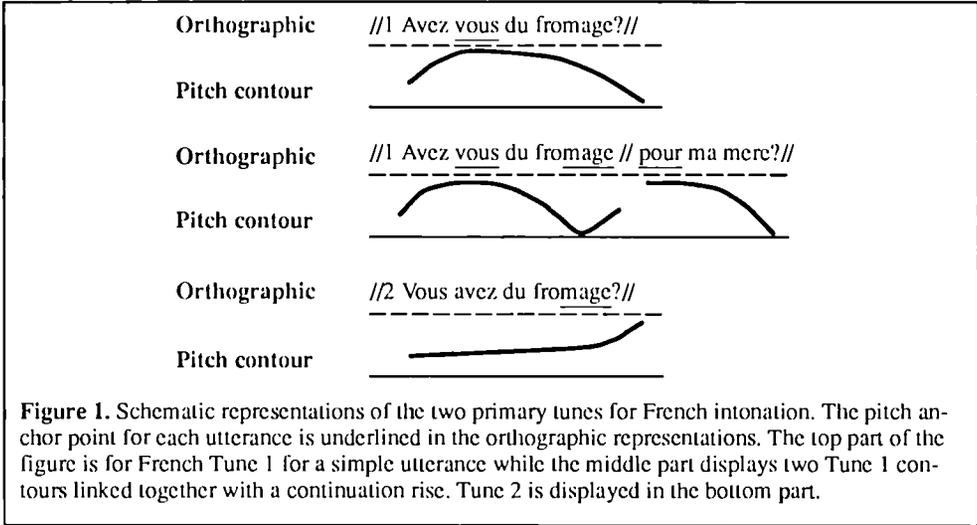
In this analysis, particular attention is drawn to two notable features which characterize intonation contours. The first feature is called a *pitch anchor point*, which specifies a segmental location within an utterance (usually a syllable) that has a significant pitch event attached to it. The second feature is a *pitch trajectory*, which describes the path taken by an intonation contour between two pitch anchor points. The use of such contour features simplifies the task of teaching intonation and allows the phonological features of all three languages to be described using a common terminology.

For each language, the discussion will be limited to the two primary intonation functions which will provide significant coverage for learners: statements/wh-questions (qu-questions in French and Italian) and polar (“yes/no”) questions.

French Intonation

According to Vaissière (personal communication), French intonation is based on unitary pitch contours which are often called “tunes”. These tunes are relatively simple in structure and the choices within them are very limited. Tune 1 is used for declarative statements, qu-questions and inverted polar questions

while Tune 2 is for non-inverted polar questions. For simple utterances, the overall contour for Tune 1 consists of a rise-fall pattern: the contour begins at a mid pitch level, rises to a high level located in the first lexical word (the only pitch anchor point in this tune) and then falls to a low level by the end of the utterance. For more complex utterances where Tune 1 contours are concatenated, there are two pitch anchor points since a small continuation rise occurs at the end of each contour but the last. Tune 2 starts at a medium pitch level, with a pitch trajectory towards a medium-high level anchor point, and then rises rapidly on the last syllable to a high level. Figure 1 displays schematic representations for the two French tunes to be taught as part of SPELL prosodic features.

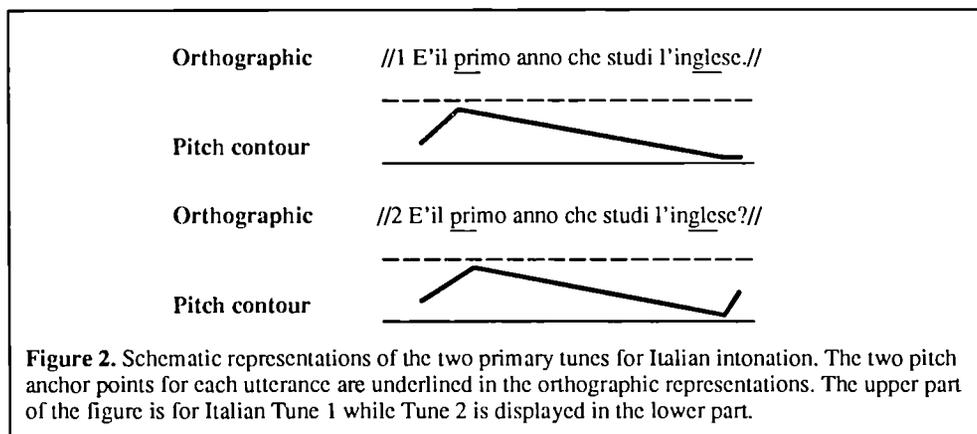


Italian Intonation

According to Chapallaz (23, 24), a tune analysis is also appropriate for Italian intonation. If these whole contours are examined in further detail then they can be described in terms of pitch anchor points and trajectories. Tune 1 is the usual intonation for statements, qu-questions, commands and exclamations. Tune 2 is the typical intonation for short, introductory non-final statements and polar questions; for these utterances, intonation is the only marker of interrogativity, since Italian, unlike French and English, does not have inverted question forms or special lexical markers. These two tunes are generally similar in structure and differentiated by the final movement of the contours. Both tunes are broadly shaped by two pitch anchor points. The first anchor point is located in the first stressed syllable of the utterance at a high pitch level while the second occurs on the last stressed syllable at a low level. The pitch trajectory falls evenly between these two anchor points. Any unstressed syllables before the first anchor point form a rising trajectory from a medium pitch level. The two tunes are differentiated by the final trajectory of the contours. For Tune 1, the contour remains at the low pitch reached by the falling trajectory at the second anchor point. The Tune 2 pitch trajectory rises sharply after the second anchor point. Figure 2 displays schematic representations for the two Italian tunes to be taught as part of SPELL prosodic features.

English Intonation

Of the three target languages, English exhibits the greatest complexity for the structuring of intonation. That is, the whole contour can be divided into a chain with choices to be made at various locations. The phonology of English intonation has attracted a considerable body of recent research. A classic (and for SPELL's purposes, a very usable) treatment is by Halliday (11). According to Halliday, there are three phonological systems at work in the intonation of English, namely tonality, tonicity and tone. *Tonality* is the system of options for dividing the stream of speech into units of intonational structure called tone-groups. *Tonicity* is the system of options for the location within the tone-group of the syllable receiving



the most prominent pitch-movement (the *tonic* syllable). *Tone* is the system of choices of the type of pitch pattern over the tone-group up to and including the tonic syllable. The stretch of speech within the tone-group leading up to the tonic syllable is called the *pre-tonic*. The stretch of speech after the tonic syllable is called the *post-tonic*.

Within a single tone group, British English evidences a large variety of primary and secondary tones as well as numerous associated pre-tonic contours. For the SPELL project, Halliday's primary Tones 1 and 2 have been selected since they provide a substantial coverage of intonational uses within English. Tone 1 is used for declarative statements, wh-questions and imperatives. It consists of a falling pitch trajectory which originates at a high level anchor point and terminates at a low level anchor point. Tone 2 is used for polar questions and certain other attitudinal information and consists of a rising or falling/rising slope. For the SPELL project, Tone 2 will consist of a rising slope which originates at a low level anchor point and rises to a high level anchor point.

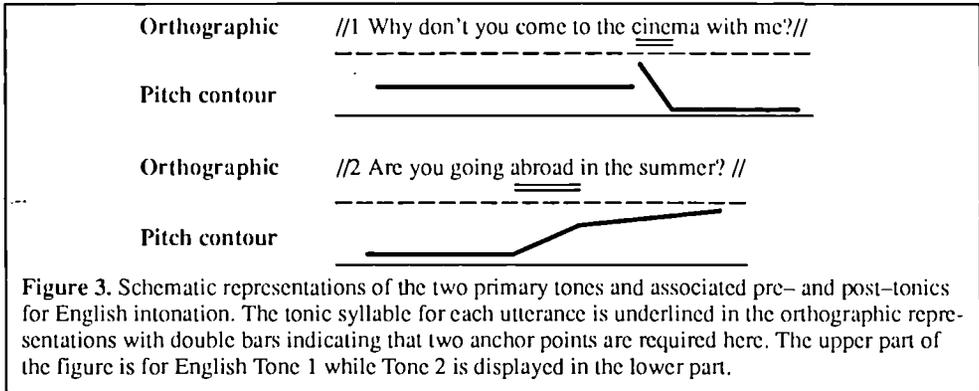
The choice of pitch contour shown by the pre-tonic stretch is contextually somewhat limited by the nature of the choice of tone. Tone 1 has the widest choice of different pre-tonic patterns that can precede the tonic syllable. In order to simplify the teaching task, one set pre-tonic contour will be taught to the student learning English. In the case of Tone 1, the pre-tonic will be a relatively flat pitch trajectory anchored at the mid level of the speaker's speaking pitch range. The pre-tonic for Tone 2 will be a relatively flat pitch trajectory anchored at the low level of the speaker's pitch range. The post-tonic choice of pitch pattern is completely prescribed, in that a tonic choice with a low terminal anchor point can only be followed by a low level post-tonic. Any tonic with a rising, non-low terminal tendency can only be followed within the same tone-group by a post-tonic pattern that continues the rising tendency. Figure 3 displays schematic representations for the two English tones to be taught as part of SPELL prosodic features.

Stress and Rhythm

Stress and rhythm are considered together here because, in English and Italian at least, they are intimately connected.

In most European languages, including English, French and Italian, stress or salience is marked acoustically by modulation of fundamental frequency, intensity, duration and segmental features. The way in which it is marked depends partly on the type of stress involved: *lexical stress* functions within a word to convey semantic or syntactic differences (e.g. English "concert" /'kɒnsə't/ (noun) versus /kən'sət/ (verb)), while *rhythmic stress* conveys the rhythmic structure within an utterance as a whole.

Most work done on "stress" in the literature examines lexical stress, and the relative contribution made by the different acoustic parameters. Control of rhythmic stress is possibly of more significance for foreign language learners. It is particularly important for English and Italian, since the occurrence of stressed syllables is one of the factors which gives them their characteristic rhythm. In addition, certain



stressed syllables are important anchor points for the pitch movements on which intonation depends. In both languages the differences between stressed and unstressed syllables are quite marked, although it is difficult to define the exact relationship between the underlying linguistic structure and measurable acoustic parameters. French lacks the apparently regular recurrence of stress beats which characterizes the other two rhythmic systems, and the distinction between stressed and unstressed syllables is not as marked (26).

The difference between languages which make rhythmic use of stressed syllables and those which do not has been formalized into a theory of speech rhythm which sees all languages as belonging to one of two types: stress-timed languages, in which the intervals between stressed syllables are controlled, and syllable-timed languages, in which control is concentrated on syllable durations (e.g. 27, 28). The theory of stress timing has been used as a teaching device for English, learners being encouraged to maintain an equal interval of time between stressed syllables no matter how many unstressed syllables come between. However, the empirical basis of the theory has been called into question by a large amount of experimental work which has failed to find any objective equality either of inter-stress intervals in "stress-timed" languages (e.g. 29, 30) or of syllable durations in "syllable-timed" languages (e.g. 31). In addition, there appear to be a large number of languages which do not fit neatly into either category (e.g. 32).

A more useful approach to rhythm appears to be emerging from these studies. It has been suggested that the perception of "stress-timing" and "syllable-timing" actually owes more to other factors in the languages under consideration, such as syllable structure, the nature of stress and the use of vowel reduction (33, 29, 30). According to Dauer (30), languages which have been classified as "stress-timed" have a greater variation in syllable length and a greater variety of permitted syllable structures. They also permit a greater degree of vowel reduction: in English, for example, unstressed vowels are typically reduced to /ə/ or /ɪ/, while in French, the only permitted reduction – of "e-muet" – typically leads to the loss of the whole syllable. Stress-timed languages also tend to have full lexical stress. Dauer proposes that languages should therefore be considered as being more or less "stress-based" according to their tendencies on parameters such as these.

This leads to a convenient classification for the purposes of the SPELL project, and one which is more amenable to automatic processing than that provided by adherence to a strict stress-timing/syllable-timing approach (see Figure 4). Thus, English is at one extreme of the "stress-based" scale: it marks the distinction between stressed and unstressed syllables quite strongly, typically with changes in the duration of the stressed vowel and the location of a pitch movement in the intonation contour, while the quality and duration of unstressed vowels are reduced (that is, the vowels have been centralized). Italian, while also strongly stress-based in that it marks stress strongly with duration and pitch, does not centralize its unstressed vowels, and has a perceptibly different rhythm from that of English. French, which is placed towards the bottom of the stress-based scale by Dauer, minimizes any durational or qualitative difference between stressed and unstressed syllables. Indeed, the perception of stress by native French speakers is notoriously unreliable (34), and many linguists deny that French has word accent at all. In addition, the absence of vowel reduction produces a rhythm entirely different from that of Italian and English.

sal]. The second area of research is to find those acoustic properties of vowels which prove to be speaker-independent. The system will need to represent, in terms of a speaker's own native vowel system, any non-native vowel for which there is no direct correspondent.

Identification of Errors in Foreign Pronunciations of Vowels.

This section discusses the most common errors found for the non-native pronunciation of English, French and Italian vowels.

English Vowels

The tense/lax high vowel contrasts /i ~ ɪ/ and /u ~ ʊ/, present in English, cannot be made by Italian and French speakers. The failure to distinguish these vowels often leads to confusion since there are several minimal pairs based on this distinction (e.g. the pair "beet" and "bit"). Though the extension of the tense-lax distinction to other vowel pairs is somewhat controversial, other pairs worthy of consideration are /ɛ ~ œ /, /ɑ ~ ʌ / and /o ~ ʌ /.

The tense/lax pairs mentioned above also include some vowels which are not present in the Italian and French vowel systems (namely, /ɪ, œ, ʌ, ɑ, ʊ/). Training on the tense/lax pairs would also be appropriate, therefore, for the teaching of the pronunciation of these new vowels.

Note that Italian and French have only one low vowel /ɑ/ which is centrally located (that is, no back/front distinction is made for low vowels). However, the English vowel system has two low vowels /ɑ/ and /æ/, the former having [+back] and the latter having [+front].

French Vowels

There are a number of problems for non-native speakers pronouncing French vowels, namely, the need to produce nasal and front rounded vowels as well as avoiding the tendency to diphthongize pure vowels.

Italian and English do not have vowels corresponding to the nasal vowels / \bar{e} , \bar{a} , \bar{o} / of the French vowel system. Speakers of these languages tend to produce a nasalized vowel followed by the extra nasal /n/. A possible training paradigm for this problem would consist in pronouncing pairs of non-nasal/nasal vowels such as / \bar{e} ~ \bar{e} ̃/, / \bar{a} ~ \bar{a} ̃/ and / \bar{o} ~ \bar{o} ̃/, which only differ in the position of the velum during articulation.

English and Italian lack the use of contrastive lip rounding in front vowels, and speakers therefore have difficulty achieving the correct lip position for the French front rounded vowels /y, ø, œ/. Training for lip position will use pairs of the French rounded and unrounded vowels /y ~ ɪ/, /ø ~ e/ and /œ ~ ε/ which differ only in the degree of lip rounding or spreading.

In the case of native English speakers, the problem of diphthongization must also be considered. The nearest English equivalents to the French pure vowels /ø, œ/ are the diphthongs /æɪ, oʊ/. While not classified as diphthongs in English, the vowels /ɪ, u/ also tend to be diphthongized in English and this habit is carried over to English speakers' pronunciation of the corresponding pure vowels in French.

Italian Vowels

There is little problem for French speakers learning Italian since all Italian vowels have correspondences in the French vowel system. In the case of native English speakers, the main problem is the diphthongization of pure vowels, as mentioned in the previous section.

Vowel Coarticulation and Vowel Normalization

Two problems arise when representing vowels by means of acoustic parameters. *vowel coarticulation* and *vowel normalization*.

Vowel Coarticulation

The phonetic context in which a vowel is spoken has a major influence on its articulation. This *coarticulation* effect gives rise to a range of different formant frequency values for that vowel within the produc-

tion of one speaker, and may cause the acoustic parameters of two different vowel–phonemes to overlap. This causes considerable problems for vowel representation, since a given formant pattern cannot then be identified uniquely. Analyses have been carried out to find parameters which might be invariant with respect to the phonetic context (35, 36), but no useful alternative to vowel formant frequencies has been found.

Vowel context must therefore be held constant when vowels are being compared. This is to be achieved by the use of minimal pairs, where the consonantal context in which the vowels are embedded is identical.

Vowel Normalization

There are two problems to be considered when comparing the same vowels produced by different speakers. Firstly, the same vowel phoneme may have a different acoustic realization for two speakers, owing to the differences in their vocal tract shape and dimensions. Thus, a given formant frequency pattern, measured in absolute terms, may be identified with one vowel in the speech of one person but with a different vowel produced by another. Some form of *between-speaker* vowel normalization is therefore required before the vowel spaces of two speakers can be compared. Secondly, *cross-language* normalization is also required owing to the multi-lingual nature of the SPELL project.

Between-speaker vowel normalization can be achieved by considering the normalized bark-scaled values using the first three formants plus the fundamental frequency, F1–F0, F2–F1, F3–F2, as suggested by Syrdal and Gopal (37). Acoustic analyses have shown that vowels are adequately represented by these parameters in American English (37) and Italian (38). The F1–F0 dimension is associated with the distinctive feature *high—low*, and the F3–F2 and F2–F1 dimensions with the feature *front—back*. As an example, Figure 6 shows the representations of all the Italian vowels in the F3–F2 versus F1–F0 plane for 13 male and 11 female speakers (Di Benedetto and Flammia, 1990). The parameters appear to normalize vowel differences across speakers successfully, though some overlap between vowel areas remains.

An alternative method of normalization being considered involves obtaining a representation of the speaker's peripheral vowels /i, a, u/ during a limited training phase (possibly covert). This could be used to locate the speaker's entire vowel space using the dimensions of F1/F3 versus F1/F2 (see, for example, 39).

Cross-language normalization presents a more complex problem, with two major issues. The first issue is whether a speaker uses similar formant frequency values for a foreign vowel which corresponds to a vowel of his or her native vowel system. In a SPELL micro feature pilot study, results from an acoustic analysis on isolated vowels of Italian and French uttered by a bilingual speaker (who did not know the purpose of the experiment) suggest that the Italian vowels have similar F1 and F2 values to those characterizing the corresponding vowels in French. The second issue is whether it is possible to predict the location within a speaker's vowel space of a vowel which does not exist in his or her native language. This question is now under investigation.

5. SUMMARY

This paper has presented the initial work completed by the SPELL project. Firstly, a basic description of the SPELL workstation was given: it is intended to be a teaching device aimed at intermediate ability foreign language learners with audio and visual aids being used to help students improve their general intelligibility within a basic teaching paradigm called DELTA. The two major phonetic areas of interest being covered by SPELL include macro (prosodic) and micro (segmental) features. A practical approach has been taken for teaching prosodic features of intonation, rhythm and stress. Whole intonation contours are broken down into easily teachable sub-units based on phonology where applicable. Rhythm and stress will be taught indirectly using the stress-based features of segmental duration and vowel quality. The micro features research is focused on the class of vowels. A number of problem areas have been identified including the English tense/lax vowel distinction and French nasal and rounded vowels. In addition, diphthongization of Italian and French pure vowels will need to be avoided by native English speakers.

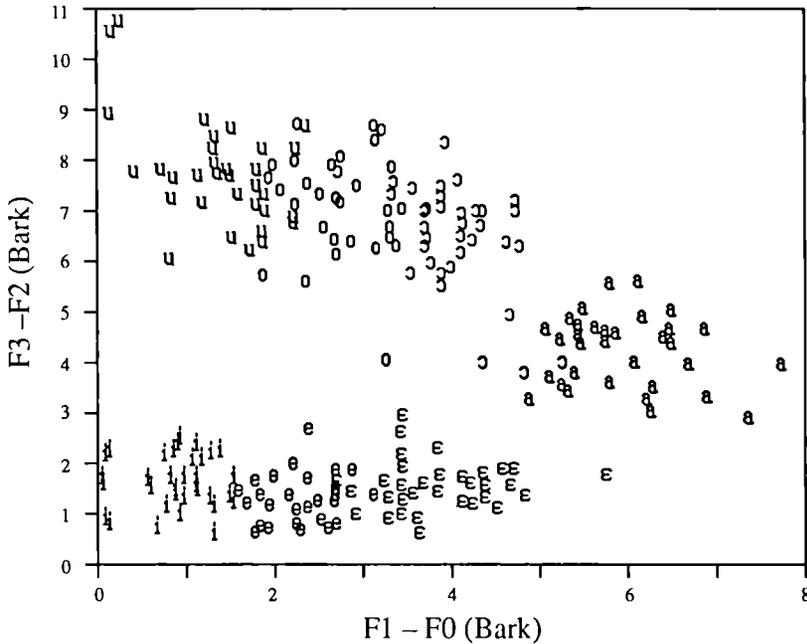


Figure 6. A representation in the F3-F2 versus F1-F0 plane of all the Italian vowels as produced by 13 male and 11 female Italian speakers. All formant values are expressed in units of Bark.

SPELL research is now being completed in a number of areas. A multi-lingual speech database has been collected and transcribed to SAM standards. Acoustic parameter extraction programs are being constructed for the features of fundamental frequency, formant frequency and segmental duration. These features are to be analyzed using a set of SPELL metrics which are under development. Of particular interest is the SPELL phonemic segmenter which will provide the segmental sequence against which the prosodic features are aligned. An important feature of this segmenter is its ability to cope with the variety of realization, systemic and structural errors produced by a non-native speaker of a language. Preliminary work has also begun on a user friendly interface for the SPELL workstation using state-of-the-art window-based graphical presentations.

The SPELL project is already considering possible extensions of this technology into other fields. The most direct extension is into the speech pathology area, particularly for speakers with articulation disorders. One example would be the assessment and rehabilitatory treatment of the speech of patients suffering from dysarthria. Another would be the use of a SPELL workstation in restoring a degree of intelligibility to the speech of patients who have undergone oral surgery for tumors of the lingual or pharyngeal structures.

REFERENCES

- (1) RIVERS, W.M. (1975); *A Practical Guide to the Teaching of French*, New York: Oxford University Press.
- (2) HARMER, J. (1983); *The Practice of English Language Teaching*, London: Longman.
- (3) MADSEN, H.S. (1983); *Techniques in Testing*, Oxford: Oxford University Press.
- (4) BERNSTEIN, J., COHEN, M., MURVEIT, H., RITSCHKEV, D. and WEINTRAUB, M. (1990); Automatic evaluation and training in English Pronunciation. *Proceedings International Conference on Spoken Language Processing*, Kobe, 1185-1188.

- (5) LEHISTE, I. (1970); *Suprasegmentals*, London: MIT Press.
- (6) ADAMS, C. (1979); *English Speech Rhythm and the Foreign Learner*, The Hague: Mouton.
- (7) CUTLER, A. and LADD, D.R. (eds.) (1983); *Prosody : Models and Measurements*, Berlin: Springer-Verlag.
- (8) O'CONNOR, J.D. AND ARNOLD, G.F. (1961); *Intonation of Colloquial English*, London: Longman.
- (9) CRYSTAL, D. (1969); *Prosodic Systems and Intonation in English*, Cambridge: Cambridge University Press.
- (10) CRYSTAL, D. (1975); *The English Tone of Voice*, London: Arnold.
- (11) HALLIDAY, M.A.K. (1973); Tones of English. In W.E. Jones and J. Laver (eds.), *Phonetics in Linguistics: A Book of Readings*, 103–126, London: Longman.
- (12) TRAGER, G.L. and SMITH, H.L. (1951); *An Outline of English Word Structure*, Norman, OK.: Battenburg.
- (13) LIBERMAN, M. (1975); The Intonational System of English. *Massachusetts Institute of Technology PhD Dissertation*, (distributed by Indiana Linguistics Club).
- (14) PIERREHUMBERT, J. (1979); Intonation synthesis based on metrical grids. In Wolf, J. and Klatt, D. (eds.), *Speech Communication Papers*, 523–526, Acoustical Society of America.
- (15) PIERREHUMBERT, J.B. (1980); The phonology and phonetics of English intonation. *Massachusetts Institute of Technology PhD dissertation*, Cambridge, MA.
- (16) FAURE, G. (1973); La description phonologique des systèmes prosodique. In Grundström, A.W. and Léon, P.R. (eds.), *Interrogation et Intonation en Français standard et en Français canadien*, 1–16, Montreal: Didier.
- (17) GRUNDSTRÖM, A.W. (1973); L'intonation des questions en français standard. In Grundström, A.W. and Léon, P.R. (eds.), *Interrogation et Intonation en Français standard et en Français canadien*, 19–51, Montreal: Didier.
- (18) MARTIN, P. (1975); Analyse phonologique de la phrase française. *Linguistics*, 146, 35–68.
- (19) MARTIN, P. (1982); Phonetic realizations of prosodic contours in French. *Speech Communication*, 1, 283–294.
- (20) KENNING, M.M. (1979); Intonation systems in French. *Journal of the International Phonetics Association*, 9, 15–30.
- (21) KENNING, M.M. (1983); The tones of English and French. *Journal of the International Phonetics Association*, 13, 32–48.
- (22) LEACH, P. (1988); French intonation: tone or tune? *Journal of the International Phonetics Association*, 18, 125–139.
- (23) CHAPALLAZ, M. (1964); Notes on the intonation of questions in Italian. In Abercrombie, D. and others (eds.), *In Honour of Daniel Jones*, 306–312, London: Longmans.
- (24) CHAPALLAZ, M. (1979); *The Pronunciation of Italian: a Practical Introduction*, London: Bell and Hyman.
- (25) MULJACIC, Z. (1972); *Fonologia della Lingua Italiana*. Bologna.
- (26) TRANEL, B. (1987); *The Sounds of French: an Introduction*, Cambridge: C.U.P.
- (27) PIKE, K.L. (1945); *The Intonation of American English*, Ann Arbor: University of Michigan Press.
- (28) ABERCROMBIE, D. (1967); *Elements of General Phonetics*, Edinburgh: Edinburgh University Press.
- (29) ROACH, P. (1982); On the distinction between 'stress-timed' and 'syllable-timed' languages. In Crystal, D. (ed.), *Linguistic Controversies*, 73–79, London: Arnold.

- (30) DAUER, R. (1983); Stress-timing and syllable-timing reanalyzed. *Journal of Phonetics*, 11, 51–62.
- (31) WENK, B. and WIOLAND, F. (1982); Is French really syllable-timed? *Journal of Phonetics*, 10, 193–216.
- (32) MILLER, M. (1984); On the perception of rhythm. *Journal of Phonetics*, 12, 75–83.
- (33) SMITH, A. (1976); The timing of French, with reflections on syllable timing. *Edinburgh University Department of Linguistics Work in Progress*, 9, 97–108.
- (34) LÉON, P.R. and MARTIN, P. (1980); Des accents. In Waugh, L.R. and van Schooneveld, C.H. (eds.), *The Melody of Language*, 177–185, Baltimore: University Park Press.
- (35) DI BENEDETTO, M.G. (1989a); Vowel representation: some observations on temporal and spectral properties of the first formant frequency. *J. Acoust. Soc. Am.*, 86, 55–66.
- (36) DI BENEDETTO, M.G. (1989b); Frequency and time variations of the first formant: properties relevant to the perception of vowel height. *J. Acoust. Soc. Am.*, 86, 66–77.
- (37) SYRDAL, A.K. and GOPAL, H.S. (1986); A perceptual model of vowel recognition based on the auditory representation of American English vowels. *J. Acoust. Soc. Am.*, 79, 1086–1100.
- (38) DI BENEDETTO, M.G. and FLAMMIA, G. (1990); Vowel distinction along auditory dimensions: a comparison between a statistical and a neural classifier. *Proceedings of the Int. Conf. on Speech Technologies*, Rome, Italy, 248–255.
- (39) MINIFIE, F.D. (1973); Speech acoustics. In Minifie, F.D., Hixon, T.J. and Williams, F. (eds.), *Normal Aspects of Speech, Hearing and Language*, 235–284, New Jersey: Prentice-Hall, Inc.

Neurocomputing

Bernard Angéniol, Françoise Fogelman, Erik Marcadé,
Jean-Michel Pimont Mimetics
Philippe Giry, Catherine Simon Thomson-CSF
Ulrich Ramacher Siemens
J.B. Theeten, P. Friedel, S.Makram Philips
P. Treleaven, M. Recce U.C.L.

Abstract:

The GALATEA project aims to promote the application of neural networks by European industry, and to develop European "standard" hardware and software tools for development and execution of neural networks applications.

The major objective is to produce a general purpose neurocomputer (GPNC) based on an open architecture, and including two dedicated neural networks hardwares and a software environment, the Neural Network Programming System (NNPS).

The second objective of GALATEA is the development of a "Silicon Compiler", a tool to generate, at relatively low cost, Neural Networks ASICs for the execution of applications developed with the NNPS.

Finally, three applications are developed in the project, all of them in image processing: Surface Mounted Devices manufacturing, Oranges automatic videograding and Optical Character Recognition, the two first ones being used for testing the GPNC, the third one for testing the "Silicon Compiler".

GALATEA started in January 1991, for a duration of three years.

1. Introduction

In the last five years, there has been a dramatic explosion of interest in neural computing, covering neural applications, neural network models, neural programming environments and neurocomputers.

The GALATEA project aims to construct a general neural computing system for Europe. This system will encompass:

- a general purpose neurocomputer hardware (GPNC), with efficient support of a wide range of neural networks algorithms,
- a sophisticated neural programming environment, the Neural Network Programming System (NNPS), allowing the efficient use of systems comprising the GPNC, domain specific processors, and ASICs, plus conventional parallel computers and workstations,
- a "silicon compiler" for rapid and low cost production of ASICs.

Three industrial applications in image processing: Surface Mounted Devices manufacturing, Oranges automatic videograding and Optical Character Recognition, are also developed in the project, the two first ones being used for testing the GPNC, the third one for testing the "Silicon Compiler".

The GPNC will be a heterogeneous machine based on a dynamically reconfigurable interconnection network, with a workstation hooked to it, hosting the NNPS. It will include two Neural Networks boards including Neural chips, memories and communicator, one developed by Philips with memories on-chip, the other developed by

Siemens, with memories off-chip; these boards will be designed for processing the learning phase.

The GPNC will also include an Interface board, including a communication processor and a low level communication driver. The architecture of the GPNC will be open, so that other boards, such as signal processor boards, other existing Neural Networks boards or sensors can be hooked to the GPNC. To allow this, and to facilitate the integration of the various parts of the project, a Virtual Machine and a Generic Board, specifying the interconnections of hardware and software have been designed.

The Neural Networks Programming System comprises five major parts:

- (i) the Graphic Monitor for controlling and monitoring a network simulation,
- (ii) the Algorithm Library of common neural networks algorithms, written in N,
- (iii) the high level neural programming language N, close to C + +,
- (iv) the intermediate level network specification language VML (Virtual Machine Language),
- (v) the compilers to the target machines which include classical workstations, the GPNC, and some other parallel machines.

The NNPS has been developed using the most common standards, C, C + +, Unix, X-window; this should allow easy integration of Neural networks developments made using the NNPS with classical software within future applications.

The "Silicon Compiler" is a Design environment starting from a high level specification using the language N (with some constraints) and leading to an optimized Neural network on silicon. The idea is to assembly building bricks of hardware on a chip in the same way neurons are assembled in a formal architecture. So, given a formal neural architecture, an optimal silicon architecture will be chosen, and the formal architecture will be folded, chopped and mapped to the silicon architecture. Good performances are expected because the basic silicon neuron will be manually optimized. The "Silicon Compiler" should allow rapid production of relatively low cost Neural Networks ASICs for the execution of applications developed with the NNPS. Although it is theoretically possible, it is not intended to give these ASICs learning capabilities, the priority being given to fast execution.

The three applications developed within the Galatea project are:

- Surface Mounted Devices manufacturing: before an automatic mounting of Surface Mounted Devices, the goal is to check defects of components, and to support the placement procedure by vision to meet the positioning accuracy requirements; big databases are available, and better accuracy and reliability are expected using the Neural Networks technology;
- Oranges automatic videograding: the goal is here to sort automatically oranges in different qualitative classes using a vision system; the main difficulties are the identification of parts of fruits non visible on views taken at different angles, and the differentiation of fruits' stems and defects;
- Optical Character Recognition: the goal is to make a system able to read printed documents as to integrate them into any word-processing system, with all conditions of page layout, type or size of font; a neural chip will be made, using the Silicon compiler, for this application.

The two first applications will be used for testing the GPNC, the third one for testing the "Silicon Compiler".

The GALATEA project brings together many of the leading neural computing research groups from European industry, research institutes and universities:

Thomson-CSF (F) Prime Contractor

Philips (NL, F)

Siemens (D)

Mimetics (F)

SGS-Thomson (I)

INPG (Institut National Polytechnique de Grenoble) (F)

UCL (University College London) (UK)

FIAR (Informatica Sistemi) (I)

CRAM (Consorzio per la Ricerca in Agricoltura nel Mezzogiorno) (I)

CTI Computer Technology Institute (GR)

INESC Instituto de Engenharia Sistemas e Computadores (P)

The GALATEA project started on January 1st, 1991; it is a three years project; the total cost of the project, half funded by the ECC, is 17 Millions Ecus.

2. The General Purpose NeuroComputer (GPNC): the integration

The integration part of the project intends to define a generic distributed heterogeneous architecture of which Neural Network boards will be a part. This generic architecture is intended to give the ability to build dedicated distributed systems integrating neurocomputers and other modules. A specific realisation of such a system integrating both boards from Siemens and Philips as well as other processors will be built and demonstrated in order to prove the coherence and efficiency of the global design.

The integration part of the project brings together all the companies involved in Neural Network hardware (Philips, Siemens) as well as companies or universities focused on the programming environment (Mimetics, UCL), thus providing a smooth and efficient integration of the different hardware and software modules. Thomson-CSF/DOI, coordinating the integration, deals more precisely with system software (operating system, communication protocols) and with generic architecture realisation.

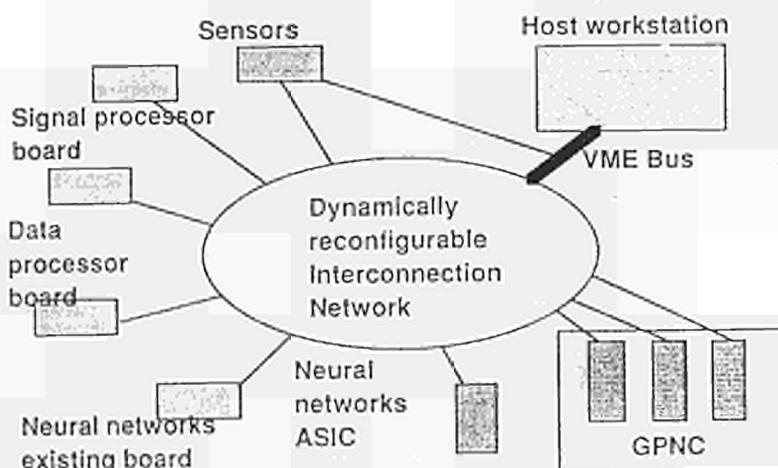


Figure 1 The Neurocomputer

2.1 An open architecture

The GPNC will be a heterogeneous machine based on a dynamically reconfigurable interconnection network with a workstation hooked to it. This architecture has to be convenient to build specific heterogeneous systems based on neurocomputing operators as well as other processors such as sensors, digital signal processors, general purpose microprocessors, ... In this way, a complete approach could be offered at the system level for problems where neural techniques are only one part of the processing.

To allow the integration of the various hardware parts as well as to keep the architecture opened, a way of describing a common interface by which such a variety of processing entities may participate has been defined: the Virtual Machine.

Moreover, instead of developing a specific compiler downwards from the Neural Network Programming System onto each dedicated board, the concept of generic board has been developed.

Finally, the GPNC will also include Interface Boards including a communication processor and a low level communication driver.

2.2 The Virtual Machine.

The Virtual Machine will be placed and structured as depicted in Figure 2, below:

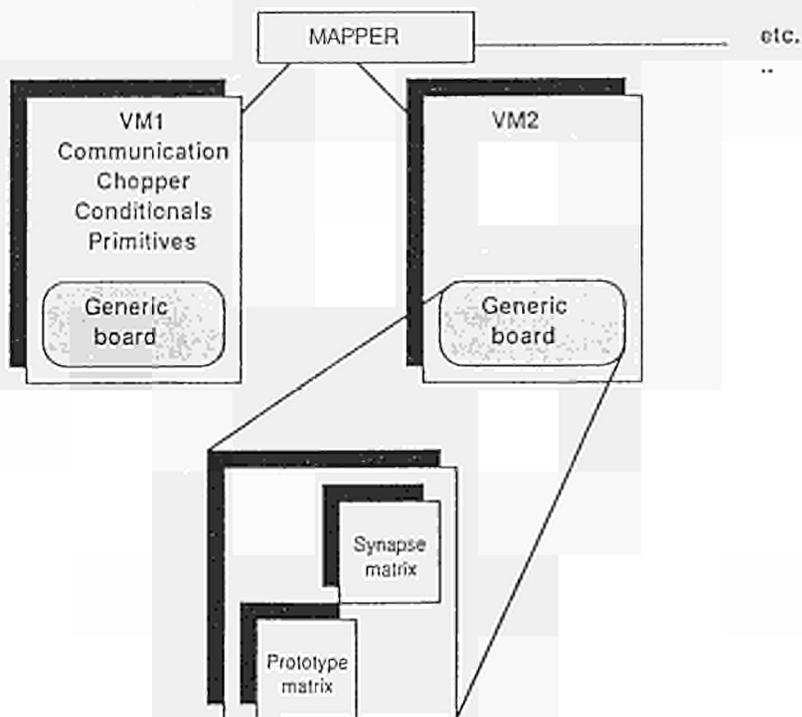


Figure 2: Virtual Machine

The Mapper maps an Intermediate Level Language representation obtained from compilation onto Virtual Machines present in the system. The Virtual Machines are

defined by a communication protocol, a set of conditionals, a set of primitives and a chopper.

The Intermediate Level Language is the output of the compilation of the high level language into "generic" primitives present on the "generic boards".

The set of conditionals enables conditional branching and loops to be effected by the Virtual Machine.

The set of primitives enables functional computation primitives to be executed by the Virtual Machine. All Virtual Machines must be able to run the whole set of primitives. However, some primitives may be done on a specialised board within the Virtual Machine. This board is called a generic Board.

The chopper should distribute generic $n \times m$ matrices type of problems on the specific hardware on the boards present in the Virtual Machine. If for instance, a physical board can implement 64×32 synapse matrices, then the problem with 400×200 matrices has to be "chopped" into banks of 64×32 .

2.3 The Generic Board.

The term "Generic Board" is a shortcut to describe the common facilities provided by all the dedicated boards and made available to the machine. The specific features of the hardware are thus hidden and the control and communication with the board brought at a common level. It is expected that the non neural boards will be integrated in this scheme by the instruction set of the board. Part of the Generic Board may be present as specific (optimised) hardware. Any primitive not present as hardware shall be emulated as part of the Generic Board in the Virtual Machine. In the long term, the Generic Board should increase in size to that of the Virtual Machine.

This supposes that a set of primitives can be extracted from the neural algorithms, being sufficiently general to be shared by the current applications but also flexible enough in order to follow the progress made in the field of algorithmics. Among them, there could be the matrix vector dot product, scalar product, weighted sum or vector normalization. To manage more specific functionalities of dedicated boards, some higher level instructions can be introduced and, in this case, these primitives could be either not available on all boards or simulated on other boards.

2.4 The communication architecture and protocol.

The GPNC will include two Neural Network boards including neural chips, memories and communication, one developed by Philips, the other developed by Siemens. The GPNC will also include an Interface Board with a communication Processor and a low level communication driver.

Important correlated issues will be considered:

- the topology of the communication architecture and its reconfigurability,
- the communication strategy (eg. protocol interface, control),
- the degree of parallelism,
- the achievable performance,
- the generality of the architecture extensibility.

The GPNC aims to be a fairly general architecture for the future developments in Neural Networks:

- its flexibility will allow integration of new hardware or software developments
- its genericity will allow its use for a wide range of applications and of networks
- its open architecture will facilitate integration of the developments in real applications

3.The Neural Network Programming System (NNPS)

In the ESPRIT Galatea project we will be developing a graphical programming environment for neural networks, which will support a range of platforms including neural computing hardware being developed in the hardware task.

The graphical software environment will include six key components:

- (1) An object oriented neural networks programming language called N which contains explicit parallelism primitives,
- (2) An algorithms library written in N,
- (3) Graphical tools for building and debugging N programs,
- (4) additional graphical tools for controlling neural network applications working with the hardware platforms,
- (5) An intermediate level language called VML which describes the primitive neural network operations in a model independent and hardware independent way,
- (6) a mapper which distributes the parts of the neural network application over the various parts of the hardware system.

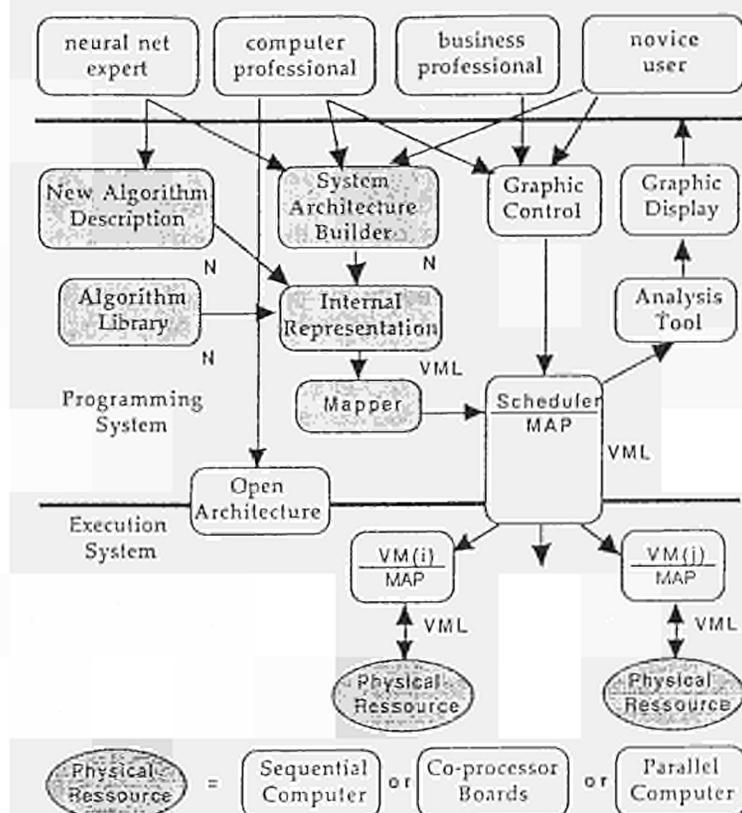


Figure 3 The software environment

The software being developed in the Galatea project is an extension of the ideas already tested in the Pygmalion project (Project 2059). The goals for each of the components of the Galatea software system will be described in detail. In each section

we will describe how these goals have been refined and shaped by the Pygmalion project.

The high level language N is an extension of the C++ language to include explicit parallelism and facilities for processing neural network models. Each of the object classes in the neural network system of N is described along with the methods performed by that object class and the connectivity of that object class. The N language provides interaction with the outside through plugs which provide the means for pattern input and output. No conventional input/output instructions per se are provided in the language, as the language expects that a main as well as overall control facilities are provided by the graphical environment. In the evolution of the N language from Pygmalion system we have adopted a more C++ like structure, and are introducing a set of type structures to handle different types of network connectivity more effectively.

In the Pygmalion project, we developed an algorithm library containing a large number of different neural network models. This was synergistic with the breadth of the Pygmalion application areas. In contrast, the Galatea algorithm library will focus on efficient implementations of a small number of algorithms. During the course of the project we will test several variations of these algorithms, with the aim by the end of the project to arrive at a very powerful small set of neural network algorithms. One set of algorithms which we will be looking at in detail are growing and pruning algorithms, or algorithms which dynamically change their number of connections and neurons.

The graphical environment in the Pygmalion project provided a very effective tool for debugging and developing neural network algorithms. It is written in C, based on the structure of the intermediate level language, uses the X windows system, and remote procedure calls (RPC) to interact with the target hardware platform. In the Galatea project we will be extending the graphical tools into several other domains. In particular we will be developing an applications builder which will allow the user to graphically put together an N program by assembling the objects with a choice of connectivity structures. Once this system is built there will be a graphical debugging system, in which there is a one to one correspondance between the instance of a window and the instance of an object in N. In the window of the debugging system each of the methods of the object will be represented by a button, and each of the parameters by a value input field.

However the requirements for the graphics in the control of the hardware and feedback from an application are different from the requirements of a graphical debugger. We are building a graphical window design tool for this purpose. With this tool the user can place in the window various types of graphical feedback, various controls of the hardware, a selection of methods buttons, and input value fields for some of the parameters of the objects in his/her application. In this manner the best way of viewing and controlling a particular application can be put together by the user.

The intermediate level language developed for the Pygmalion project contained some useful ideas, but is largely being rewritten for the Galatea project. This language was written at too high a level, and did not provide for straight forward mapping onto the hardware. In addition the hierarchical structure of the language was too rigid. With the benefit of hindsight we are building a language called VML for virtual machine language, which sits exactly between the N language and the hardware. This language contains the common primitives of neural network algorithms, but is phrased in a series of matrix operations which provides for efficient execution on hardware for matrix arithmetic. As discussed in the hardware section of this paper, matrix processors are seen as the optimal platform for heavily interconnected neural network algorithms. However, we are not restricting the VML language to hardware based on matrix

arithmetic. We expect it to execute on a wide variety of hardware platforms. The VML language describes the actions of an autonomous virtual machine, which is communicating with other virtual machines, executes part of the neural network system, and keeps a data map of the data distributed over the rest of the virtual machines.

The last key component of the software system is the mapper and the scheduler. The mapper is a compile time system for distributing the components of the neural network algorithm over a number of virtual machines. This task is made more difficult by the fact that these machines are not identical, each may be better at a subset of the computations, and the communications speed of each of the machines may not be the same. In order to best perform this task we are theoretically analysing the performance trade offs with different partitioning and different communication schemes. A simulation will be built to further test these ideas, and a manual mapping system will be put in place to provide an application specific test of the mapping efficiency.

All of these components are being developed in several stages. The first stage is after the sixth month of the Galatea project. At this time the design of each of the key components will be completed. Nine months after this point we intend to have a fully functional first pass system. This system will be characterised, and then merged with the prototype hardware at month 18. Between month 18 and month 30, six months before the end of the project, we will refine all of the key components of the system to provide the final deliverable

4. Neural networks General Purpose Hardware

Two general purpose boards dedicated to neural networks learning (the GPNC box in Fig.1) will be developed in Galatea, one by Siemens, the other one by Philips.

4.1 The Siemens board : Synapse-1

The Siemens board architecture has been designed which features the following characteristics:

- System architecture is based on the compute-bound algorithmic elementary operations shared by all neural networks.
- Systolic execution of the elementary operations by a specific VLSI Neural Signal Processor MA16, and of the remaining operations by commercially available DSPs or microprocessors.
- Carrying on the systolic computation to 1- or 2-dimensional arrays of MA16s.
- Each MA 16 is provided with its own off-chip weight memory as well as template memory.
- Systolic communication and control architecture for the array and the weight memory.
- VME interface to the host.
- Neural algorithm description language that specifies the neural algorithms in terms of a sequence of elementary operations and their optional concatenations with DSP or host operations.
- Cross-compiler that translates the description language into the machine language of the neurocomputer board.

The proposed neurocomputer concept is sizeable to the applicational domain in terms of processing power, memory and flexibility. Throughput rates at the chip site of the order of 800 MC/sec (1 Connection = 16 bit) can be realized with 1 μ m CMOS technology. At the board level a system performance of 4.2 GC/sec can be achieved

with 4x2 of these chips arranged as a 2-dimensional array. In terms of network size and support for various neural nets and learning, a neurocomputer concept of this kind gets to new dimensions. It will enable the user to access realworld applications in reasonable time and study the potential of neural networks. The neurocomputer drafted in the Figure below can therefore be considered as a research instrument as well as a design platform for working out application-specific neural system architectures in terms of dedicated software and hardware.

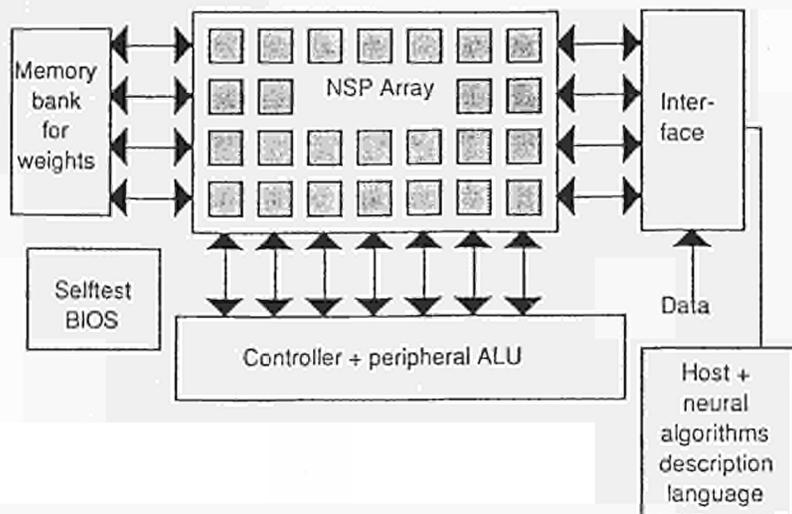


Figure 4 Synapse-1: A board that synthesizes neural algorithms on a parallel systolic engine

Goal: VLSI for fast & flexible emulation of neural networks for real-world applications

Features: Recall & learning in real-time, flexibility

Solution: Systolic synthesis of elementary compute-intensive algorithmic strings

Chip: VLSI Neural Signal Processor for execution of elementary algorithmic kernel

Board: 2-dimensional array of NSPs, distributed control & communication

4.2 The PHILIPS board: The option for small size neural networks

The knowledge that is being acquired from our experience using board based on the existing L-Neuro 1.0 and on several applications leads to the following remarks:

(1) Most applications today require a subtle partitioning of algorithms in neural and classical parts. The fast interaction between neural and non-neural processing steps is of paramount importance for performances on applications. Integration on the chip should go further than the weighted sum and the Hebb rule. Deciding which primitive is sufficiently computer intensive, apt to be parallelized and generic to a large class of problems is thus central to defining efficient hardware for neural algorithms.

(2) The control of data flow may be very time consuming and degrade performances. New neural network algorithms cannot be implemented with dataflow constructive algorithms.

(3) The neural networks used up to now in applications require reduced numbers of neurons. Typically most applications use from a few tens to a few hundreds of

neurons. It is in many cases more effective to cut the neural network in an intelligent manner (shared weights or TDNN) than to inflate the network.

(4) An open GPNC should enable to change boards or host machine in order to be able to build a board optimised for a family of applications without having to rewrite the software environment and to redefine the communication protocol.

From these considerations our approach to a board for the GPNC is the following. In a first stage we are defining an open environment that will enable to plug boards from different manufacturers as long as they can be made compatible with the Virtual Machine communication protocol and sets of primitives. This phase is well under way and serves for the definition of a common meeting point for hardware and software. It has been described in the integration workpackage.

In parallel, based on our knowledge acquired on L-Neuro 1.0 we are exploring new architectural solutions for a version 2.0 of the chip. This is done using our internal behavioural compilation language (ALMA). The demonstrator at month 18 will essentially aim at showing the viability of the concepts of the Virtual Machine and the Generic board. It may be used on DSP or L-Neuro 1.0. A decision will be taken after the first Generic Board prototype has been built, as to whether we integrate this chip into the final board for Galatea or another existing chip that is well adapted to our approach of neural networks.

There are two main interests in developing two boards in Galatea; one is that the two boards are complementary, the Siemens one being more targeted towards big networks requiring lot of resources for learning, the Philips one addressing rather applications made by cooperation of smaller networks; the second reason is that it was necessary to test the capability of integration of several boards to check the architecture.

5. The "Silicon Compiler": a Neural Networks ASIC generator

The objective of this workpackage is to provide a design environment starting from a high level specification and leading to an optimized artificial neural network on silicon. This environment can be viewed as a Silicon Compiler for Artificial Neural Networks (ANN). Starting from a high level description, the synthesis software proposed here uses as basic tools, block generators or classical synthesis tools of commercial silicon compilers. The generated dedicated ASIC will then have good performances/development price characteristics.

It must be well understood that the objective is not to have a general silicon compiler scheme but to be able to develop cheap hardware corresponding to neural network applications. In this view, an architecture must be selected to have efficient silicon compilation and the architecture developed in Pygmalion seems a very good candidate.

The starting point is the high level specification language called N defined in the former project Pygmalion and improved in Galatea. In this language, the user will be able to define its application in terms of behaviour of neurons (Algorithm description) and interconnects between these neurons (Application description). Most of the time, the resulting neural network will be trained using the simulation environment, eventually using general purpose hardware accelerators (like the GPNC). Most of the currently known training processes require a great accuracy especially in the definition of the weights on the interconnects that must be implemented in floating point format on single or double precision. On the opposite, all implementations on silicon will require to store the weights in very few bits.

This is why the silicon compiler workpackage does not intend to generate silicon design for chips used in the training phase but simply for the relaxation phase. In this case, work has been done proving that the global performance of the network is not significantly reduced when reducing the precision of the activities of neurons and weights of the interconnects.

The N library neuron processors will be described in a standard format developed during this project, they will be parameterized in terms of bits of the processing units and weights, this format is called the resource format and will be defined by INPG. The first module specific to the Silicon Compiler is a Type Simulator (TS) developed by Mimetics, allowing the user, from the simulation environment, to test performance degradation when storing the weights with a reduced number of bits and computing within each neuron the relaxation process with a reduced number of bits for neuron activity.

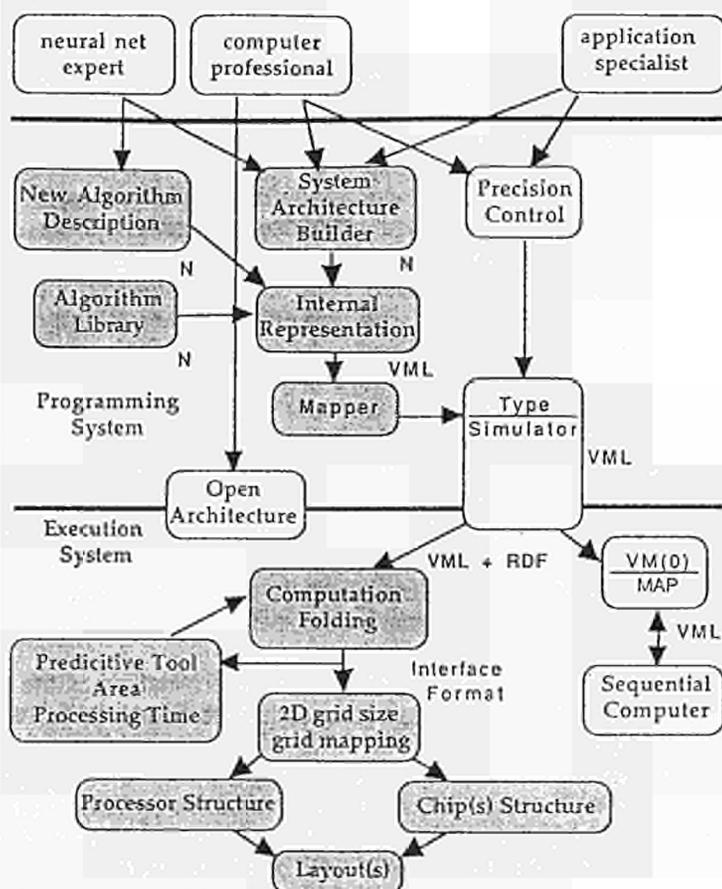


Figure 5 The Silicon Compiler

After this tuning, neurons descriptions will be linked to specifications of block generators and synthesis tools of silicon compilers commercially available for a given

technology. Two silicon compilers could be used here, the VLSI Technology Inc. as it represents the most advanced one on the market and the UNICAD ST one. This will be done inside a mapping process aiming at downloading the application into a reasonable Silicon architecture (in terms of number of chips, number of neurons per chip etc...). This mapping process, which is a very complex task, will be realized by INPG and UCL, and will be tested on small architectures as well as on the OCR workpackage architecture. The main problem to be solved at this stage is the fact that it is not thinkable to have one physical neuron per formal neuron defined in the application (as an example, the OCR application should use around 3000 neurons, and 60000 connections) which means that each physical neuron will play the role of several neurons of the application. This mapping process will be cut into two phases called mapper and chopper for problems related to the folding between chips and internal to each chip respectively.

It must be emphasized that the main goal of the Silicon Compiler workpackage is to provide generic tools, but the first "near-commercial" application of these tools will be the OCR application.

6. Industrial Vision Work-Package : Two test applications for the GPNC

6.1 Industrial Vision for Surface Mounted Device Technology

Philips I&E Division manufactures automated systems for high speed mounting of (SMDs) Surface Mounted Devices (typically 36000 components per hour). Although the yield is very close to 100% for every mounting step, the large number of components on a board brings down the fabrication yield so that correcting interventions are frequently required.

Industrial vision can provide an efficient solution to this challenging problem, The different tasks involved in this SMD applications are:

Task 1: where Philips (I&E and CFT) is responsible and which aims at providing a hardware platform both for demonstration purposes and for the generation of the necessary image database. This platform will integrate the results of other tasks and test them under realistic conditions.

Task 2: where Philips (I&E and CFT) is responsible, aims at generating the image database and at establishing image format standards among the partners.

Task 3: where FIAR/IS is responsible deals with automatic visual inspection SMD components before placement. This task has been decomposed into three subtasks as follows:

3.1. Component Identification together with determination of the component presentation of pin 1 (one among 4 for a square component). To be carried out by Thomson LER.

3.2 Once the results of the previous subtask are known. the angular position of the component in the feeder should be determined. To be carried out by FIAR/IS.

3.3 Once the results of the above subtasks are known, component damage, if any, should be detected. To be carried out by Phi1ips/LEP.

Task 4: where Philips/LEP is responsible deals with board inspection before component placement. The main aims are to determine the position of the relevant footprints and to determine whether or not the glue and solder paste have the correct position and volumes.

Task 5: where CTI is responsible aims at developing three dimensional component placement strategies through the detection and accurate localisation of reference

features on components and on the board. The three dimensional information can be made available by different means. For example, it may be extracted from pairs of stereo images or from image sequences.

Task 6: where Philips/TFP is responsible deals with postplacement inspection. Correct component positioning on the board and the correct paste appearance should be checked for.

Task 7: where FIAR/IS is responsible will compare the performances obtained with the neural net approaches used in the previous task with well established classical approaches.

For the SMD application, effort has been concentrated on preparation of the hardware platform and of the image data bases (Task 1 and 2). The partners are also busy in preparing suitable simulation tools.

6.2 Automatic Videograding of Oranges:

This application is the responsibility of CRAM.

The aim of this task is to find neural net solutions to critical steps in the fruit videograding process. Three orange grades have to be defined. The third grade is that of severely damaged oranges which can only be used for their juice. This grade is very easily detected. The two first grades are much more difficult to separate from each others. The criteria for separation are the type of defects (some are less harmful than others) as well as their total count on each orange.

Work is currently in progress to determine the most suitable type of sensors in order to provide sufficient discrimination between the different defect types involved.

7. Optical Character Recognition: the first Neural Networks ASIC

Optical Character Recognition aim is to scan a text bitmap and convert it to its ascii form. Several software products exist, which process a page in about two minutes, with a few to many errors. Some hardware PC boards are available, with variable increase in speed and price.

The commercial aim of the OCR workpackage is to produce a prototype of such a board, with high performances in speed and recognition. Speed performance should be achieved by the NN Dedicated Asic approach. Recognition rate should be good due to the known applicability of NN techniques in the Pattern Recognition field.

The processing chain of an OCR system includes text segmentation, recognition of individual characters, and post-processing (spell checking, page layout restoration,...). The recognition will be handled by the NN chip, but the pre- and post-processing will be performed by conventional chips.

The technical objective of the OCR workpackage is to test the Silicon Compiler. The complexity of this NN architecture, the high speed to be achieved, have already introduced new types of neurons, new ideas in neuron data flow conception.

This workpackage is under the responsibility of Mimetecs, which specifies the Neural Network and develops the software. INPG designs the chip to be produced by SGS-Thomson. Thomson-DOI designs and produces the board.

8. Conclusion

To conclude, the GALATEA project has been running since January 1991.

Already major progress has been made; concerning the integration, a real progress has been made in the definition of the virtual machine concept allowing different hardware processing entities to participate together to provide the definition of a generic heterogeneous architecture based on neurocomputing operators as well as other processors. Future work concerning the analysis of parameters about communication will allow to determine the interconnection medium between all these entities.

.. The analysis of neural algorithms has been completed and a set of compute-bound operations identified which are shared by all neural nets. VLSI design of a neural signal processor MA16 that executes these operations has started. The functional block architecture of the system has been fixed at chip and board level. Next, the specification of data and control flow between the coprocessor boards and the host will be attacked.

The design of all the key components of the Software and Silicon Compiler work-packages have been completed.

The preparatory phase for the industrial vision and OCR workpackages is completed. The most challenging problems have been pinpointed and the first neural net solutions proposed.

The global approach to the realization of compatible tools for neural networks users, in an open architecture allowing integration in applications, is unique worldwide. Our goal is to develop a set of tools widely used in Europe, but also outside Europe. It should put the European industry in a strong position for the neural networks market as well as for the neural networks applications.

ADVANCED BUSINESS AND HOME SYSTEMS - PERIPHERALS

SPRITE A SYSTEM FOR TECHNICAL DOCUMENTATION

Jirka Hoppe
Océ Wissenschaftliches Forschungsinstitut
Asylstrasse 125
8032 Zurich
Switzerland
tel: +41 - 1 - 55 00 24
E-mail: jh@owf.uu.ch

Summary

The SPRITE project aims at the development of an integrated system for the production and maintenance of technical documentation. SPRITE provides for authors and managers an information management system supporting direct access to the complete information environment, ranging from documents stored in the internal data base over paper based files (drawings, reports) to electronic CAD files and external databases.

This paper discusses important aspects of a system for technical documentation: document architecture, organisation of the document space, versioning, multiauthoring, and integration of data stored on external media. For each of these aspects, the solutions developed in the SPRITE project are presented.

1. Introduction

Writing documentation is an integral part of any technical product development. A significant amount of time is spent on describing the functionality of the product, giving insights into technical details, providing maintenance instructions, specifying marketing information, writing user manuals, etc.

In a first approach, a system for technical documentation needs about the same functionality as any other system producing documents: a powerful editor capable of handling text and graphics, a file storage, and a printer. Detailed studies of requirements of technical documentation revealed additional functionality that is not covered by conventional document processing system and that needs some additional support:

- Technical documentation is often very large. Documents having a size of hundreds or even thousands of pages are not exceptional.
- Technical documentation rarely consist of a single document; often many different documents have to be combined thus creating a large amount of documentation attached to one product. Optimum organisation of storage and retrieval of documents is crucial for the performance and acceptability of the system.
- Technical documentation has a very long active life. Every product is changed over a period of time and its documentation must follow its evolution. It must be possible to create new versions of the documentation and to access old versions describing a several year old product.
- Technical products exist often in different variations that are very similar in the common part, but differ in small details (example: the documentation of a car consists of a large part that is independent of the kind of a transmission; small parts -

describing the manual vs. automatic transmission - are, however, different). The documentation system must support the comfortable handling of such shared common information as well as handling of the different parts.

- Due to its size and complexity, technical documentation is mostly developed by a team of authors. A system for technical documentation has to support multiauthoring and functionality for the management of a group of authors.
- Technical documentation is seldom written completely from scratch. It mostly reuses already available information (e.g. paper drawings, data available on other computers). Import of such external information must be easily supported.

The goal of the SPRITE project is to develop a prototype of an integrated system for technical documentation. The SPRITE system is intended to be a demonstrator showing all essential functionalities needed to create and maintain technical documentation.

The SPRITE project was started in January 1989 with a project duration of three years and a budgeted project size of 79 man years.

This report is structured in the following way: Chapter 2 describes the SPRITE approach to solve essential problems of a system for technical documentation. In the subchapter 2.1 the structure of large documents is discussed, subchapter 2.2 describes the version model, subchapter 2.3 handles the approach taken for multiauthoring, subchapter 2.4. discusses the document storage and retrieval, and subchapter 2.5. describes the integration into existing environment. Chapter 3 discusses the evaluation of the system. Chapter 4 contains the conclusions.

1.1 Relation to existing products

Recently first systems appeared on the market having a similar goal as Sprite. The following systems may be considered as representatives of such systems: The Publisher (ArborText), DOC and PicEd (Context), Interleaf, KEEPS (Kodak), or Documenter (Xerox) (this list is by far not complete). Such systems, however, do not cover all aspects of technical documentation; especially the integration with a data base and integration of recognised paper documents are missing. A good overview of such systems can be found e.g. in [10].

Beside of the above mentioned systems for documentation, several systems for software engineering or version handling support similar concepts as targeted in Sprite (multiauthoring, versioning, etc). Among others the following systems are popular: EXODUS[4], MINOS[3], SCCS[2], RSC[9].

2. Solution of problems

In the previous chapter a number of important aspects of systems for technical documentation have been listed. This chapter describes for each such aspect which solutions are offered by the SPRITE system in order to support such functionality.

2.1 Structure of large documents

Technical documentation can be very large. Documents having a size of several hundreds or thousands of pages are no exception (rumours say that the documentation of a Jumbo Jet fills the entire Jumbo Jet). Besides text that can be stored in a reasonable dense way, documents contain often components requiring large amount of storage (e.g. bitmaps). For obvious reasons it is not possible to store such documents in a single flat file, as currently done by most word processors. Any operation on such

document would require rewriting of files having a size measured in megabytes. Such a system would be too ponderous.

An obvious solution to this problem, is to cut large documents into small pieces and to store such pieces separately in a data base. Pieces in the data base may be updated as single objects. An additional organisation is needed to collect all such pieces to one final document.

This approach has an additional advantage: it allows sharing of document contents between different versions of one document, or between copies of one document.

The organisation of documents in a data base is generally based on the ODA document model [1]. Certain deviations or simplification of this model were necessary in order to include other innovative concepts investigated in SPRITE (e.g. multiauthoring or versioning) or to keep the system implementable within the short project time.

Similarly as in the ODA model, a SPRITE document consists of four related parts:

- *logical structure* describes the structure of a document. It associates the content of a document with entities like chapters, headings, paragraphs, figures, etc.
- *content* contains the characters of the text, graphical objects, raster data, or business graphic.
- *formatting styles* determine the transformation of objects in the logical structure to objects in the layout structure and the rendition of the content.
- *layout structure* describes the geometrical position of objects on the presentation media. It associates the content of a document with entities like pages, layout blocks on the page, etc.

These four parts of the document architecture are stored generally on physically different places in the system. The *logical structure* is stored in form of tables in a relational data base. The *content* is stored either as a data part in the data base or as plain UNIX files on the disk. The content may be stored on both magnetic and optical disk. The *formatting style* is stored similarly as content on a file. The *layout structure* is not stored on any disk media. It is computed on the fly when a document is loaded in the memory of a workstation.

The *logical structure* consist of a hierarchy of logical objects - called components. Components can be further subdivided into basic and composite components. *Basic* components are elementary; they are not further structured. Basic components contain the actual contents of a document, e.g. one paragraph of text or one graphic image. *Composite* components, on the other hand, build the hierarchy in the logical structure, because they may contain, beside basic components, composite components themselves. The most important representative of a composite component is a *chapter*. A chapter consists of content and of further subchapters.

An important kind of a component is the *import* component. Such a component contains a pointer to another component in another document. This allows that e.g. a picture can be developed only once, but it is included in many documents. An important aspect of an import is, that the importing documents follow the changes of the imported document. This means that, if the imported component is changed, such changes are visible in the importing documents when they are next time opened.

Imports are used to implement the *bottom-up* multiauthoring working style described later in the multiauthoring section. Each author first generates private documents, independently on other authors. All such documents are later imported to one connecting father document.

The *content* consists of pieces of information each representing different kinds of content: a sequence of characters, graphical objects, raster data, or business graphics (diagrams).

An important requirement of the SPRITE project was to provide a functionality supporting a consistent layout of technical documents. The SPRITE document architecture supports a definition of a *style* that is common to a number of documents and that controls the formatting of documents. Such style is normally designed by an artist and it cannot be changed by a normal technical author. For exceptional situations, a support for a definition of private deviations of the common style is, however, provided. Such deviations are called *local* style and they are valid for one document only.

The following example (see Figure 1.) shows a structure of one simple SPRITE document. A document describing a *Car* has two styles attached: the GDS (global document style) contains the description of the common global style, the LDS (local document style) contains the private style changes of an author. The document has at the top level 3 chapters (1., 2., and 3.). Chapter 2. has three subchapters and chapter 3. has two subchapters. Their content is not further decomposed in the picture. Chapter 1. consists of three components: two contain directly some content, the third component is an import component that points to a component in a document *Engine*.

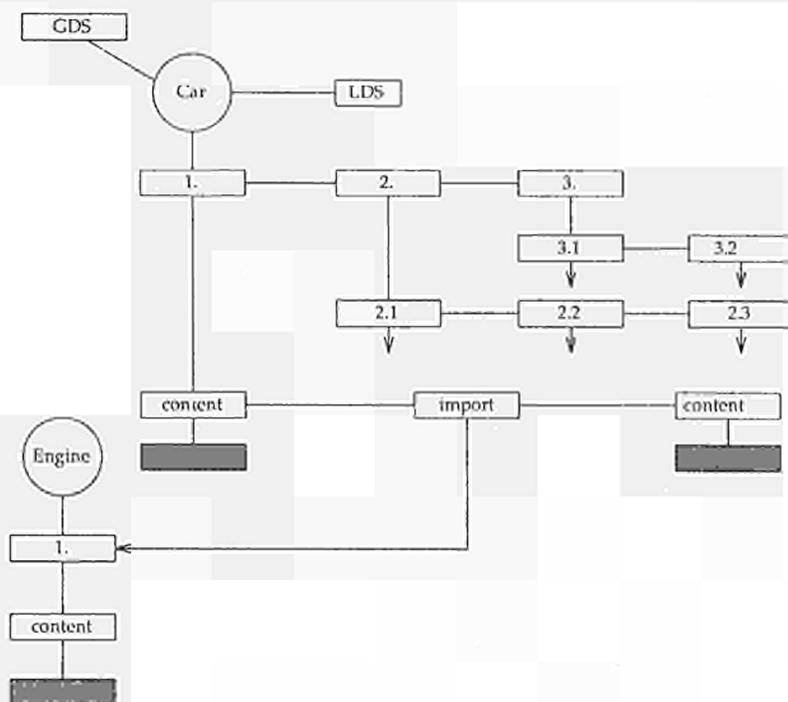


Figure 1. Organisation of document structure

2.2 Versions

The SPRITE version model of documents supports two orthogonal dimensions of versions. The first dimension covers a development of a document over time. Such

time-dependent versions are called *editions*. Another dimension of the model covers slightly different variants of the same product (a car may be delivered with either a manual or automatic transmission). Versions of this kind are called *configurations*.

2.2.1. Editions

The *edition* dimension is based on similar ideas as found in other systems dealing with versions. When a document is created, it exists as a revisable document called *checkpoint*. Such checkpoint can be edited - the results of such editing are always rewritten in the data base when an editing session is finished. A revisable checkpoint can be *frozen* - from this moment on its content cannot be directly changed any more. Editing such a frozen document causes a new checkpoint (again revisable) to be created and logically connected with its father. Since it is possible to edit either the last frozen checkpoint in the history or an earlier checkpoint, not only linear lists of checkpoints, but as well *checkpoint trees* can be created.

Important documents of a checkpoint tree may be marked as *editions*. Documents marked as editions are officially released and mostly have passed formal reviewing or verifying procedures. Similarly as checkpoints, editions are organised in a tree form.

2.2.2 Configurations

Configurations are documents which describe the same basic product, but differ in some details. For example, a *car manual* might exist in two configurations, one of which describes a car with an automatic transmission, the other describing a manual transmission option. Most of the content of these two documents is the same and can be shared, except the portions specific to the configuration.

SPRITE presents two views of configurations to its users. To facilitate creation and editing of configuration documents, an author sees each configuration document as a collection of separate documents called *building block documents*. Some building blocks contain content common to all configurations, others describe each of the configuration items. Building blocks are related to one another by a combinability relation, which defines how the building blocks can be combined to create complete documents. Each father building block document contains special components called *placeholders* that specify the exact place where son building block documents should be included.

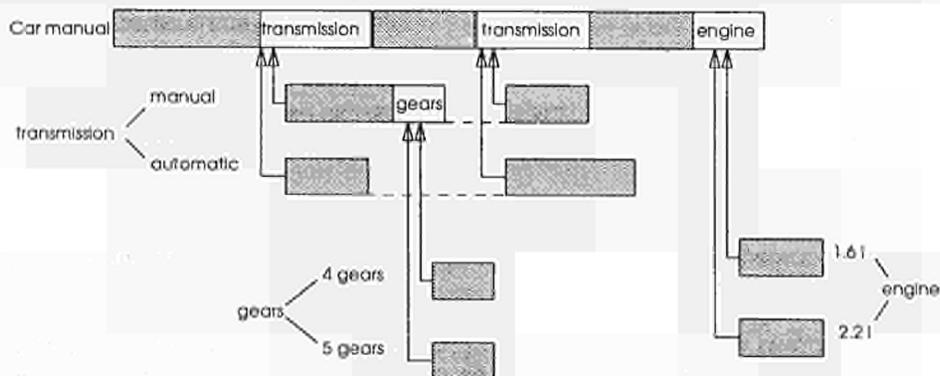


Figure 2. Example of configurations

In the above example the *car manual* describes the standard features of the car - it contains a content that is neither dependent on the transmission kind, nor on the size of the engine. This document has two placeholders: one marked "*transmission*" and one marked "*engine*". There are two "transmission" building block documents, "*manual*" and "*automatic*". Each has two components, which refer to the two placeholder components in the root building block marked "transmission". The "manual" building block has a component marked "gears", since manual configurations of the car can have different number of gears. "Gears" building blocks consist of one component which will refer to the placeholder in the first "manual" component.

On the other hand, readers of documents and final users are not concerned with how each building block document is created. They just see complete "composed" documents. A reader wants to specify a configuration of a document, like the "car, manual transmission, 5 gears, 2.2l engine", and not to be aware of building blocks and combinability. At the same time the user would like to be prevented from trying to access illegal composed documents, such as the "car automatic, 5 gears" (which is illegal since the automatic transmission does not have a choice of gears).

2.3 Multiauthoring

As mentioned previously, technical documents are often very large in size. It is clear that such documents cannot be not written by a single author, they are usually developed by a group of authors. The SPRITE system has to support the work and management of such a group of authors. An implementation of such a requirement is, in practice, rather difficult. Every organisation has its own working style and it is not easy to implement a system that supports all possible kinds of working styles. The SPRITE system concentrates therefore on two important dimensions of the work distribution as described in the next two subchapters:

2.3.1. Functional vs. structural division of work

The work on one document can generally be distributed to authors in two ways:

- In a *structural division of work* a large document is divided into structural pieces like chapters. Each author is responsible for one chapter with all its contents and all its subchapters i.e. for all text, graphic, photographs, tables, etc.
- In a *functional division of work* each author is responsible for one kind of content. Some authors write only text, some others create drawings, some others are experts in scanning pictures and correcting them.

Hybrid forms of the mentioned divisions are possible as well: An author may create both text and graphics, but scanning of pictures is still done by a specialist.

2.3.2. Top-down vs. bottom-up working style

Large documents can be developed generally in a *top-down* or a *bottom-up* working style.

- In a *top-down* working style, first a rough structure of a document is defined (e.g. the top level chapter division), and then this rough structure is filled in by several authors concurrently.

- In a *bottom-up* working style, first the constituent parts are created separately, which are later combined together to build the complete document.

2.3.3. Multiauthoring model

SPRITE provides a multiauthoring model supporting all working styles mentioned above. It gives support for three classes of users:

- *manager* supervising the common work, controlling the progress, and assigning work among to authors.
- *authors* creating and updating one document.
- *readers* reading documents.

The support for multiauthoring is based on an assignment of document parts (e.g. one chapter, one drawing, one table) to authors. The assignment of objects is provided by setting authorisation permissions to such objects. Usually one part is assigned to one author only. If more than one author has an update permission to an object, a locking mechanism guarantees secure access.

Authors may put either full documents or parts of them on their desktop. Document parts are handled then like full documents, it means such parts can be edited, printed, or handled by other applications that work with documents (e.g. browser). If an author with a write permission starts an application on such a document part, the part becomes locked and cannot be accessed by other authors. The access is always limited to that part that was specified when the application was started. In this way a secure concurrent access by many authors to one document is supported.

The two kinds of work division (structural vs. functional) are mapped into this model by assigning full chapters (for the structural division) or single components (e.g. a drawing or a table for the functional division) to one author.

The *top-down* working style is supported in this model by creating first a simple document consisting of e.g. empty main chapters and assigning such chapters to authors. The *bottom-up* working style is supported by the concept of imports described above in the document model.

During concurrent use of a document, write-write and read-write conflicts may occur. Write-write conflicts happen when two authors try to change the same objects in a document at the same time. Read-write conflicts occur when someone tries to read a document part that is at that very moment being edited. A locking schema is used to solve these concurrency problems.

2.4. Document storage and retrieval

Every organisation producing technical documentation has to manage large amounts of documents, each document being large itself. The organisation of the documentation storage providing easy retrieval and management, is one of the most crucial tasks of the system. In order to be easily understood by users, such organisation should, as far as possible, be compatible with classical methods for storing paper documents (filing cabinets, files, documents-cases, etc).

The organisation of documents is often used to keep together a collection of documents that were delivered together with one product. Such a collection consists of a large number of documents of different kinds (e.g. technical documents, sales documents, maintenance documents).

When working with large documents in the range of 1000 pages, it is necessary not only to locate entire documents, but a retrieval of the internal parts of one document is essential as well. The SPRITE system must be able to locate an specific drawing or table without scrolling through the whole large document.

Besides the plain storage of documents, users expect support in obtaining information about related documents in order to get a full knowledge about the described product.

Due to the project's limited budget, only the most important retrieval aspects have been implemented. It was not possible to include more sophisticated methods like a full text search or a full hypertext system.

2.4.1. Model of the document space

The basic objects of the SPRITE document space are *documents*. Such documents may be put in mutual relations represented by folders, version clusters, and check-point/edition trees.

The basic structure of the SPRITE document space is represented by a (mostly) hierarchical folder structure. A *folder* is a named set of folders, version clusters, and documents. Each of these objects may be contained in more than one folder. Since folders may be contained in many folders (concept similar to symbolic links in UNIX), the resulting structure is not strictly hierarchical.

The concept of having one document in different folders is especially useful to fulfil the above described requirement of keeping together a collection of documents related to one release of a product. For such a case a dedicated folder is created that keeps a kind of links to all referenced documents.

Version clusters enclose all versions of one document. Such clusters contain documents in two dimensions: in the configuration dimension and in the edition dimension. The *configuration dimension* handles slightly different versions of a document depending on configurations of the product. The *edition dimension* handles document versions that were created by updating previous editions of one document. They contain the explicit relation which document was created by editing of which previous document.

SPRITE supports the possibility to connect related documents by means of links. Such links are created by a user either to connect related documents (e.g. a technical documentation of a car and a sales brochure for the same car) or they may e.g. indicate a dependency of one document on another (e.g. an automatically-generated table of contents is dependent on the main document). A very simple version of a hyperlinks system can be defined in this way.

2.4.2. Browsing/Retrieval

Users of the SPRITE system can locate objects in the document space either by browsing (navigation) or by specifying queries.

During browsing a user sees a series of views, each describing the current environment in the document space. By selecting an object and opening it, the user proceeds to the next view.

The following scenario describes how to locate an specific picture in a description of the automatic transmission of a car CAR1 (see the example description of a version cluster above). The user starts browsing at the level of *folders*. S/he first locates the folder *manuals*, in this folder another folder *automobiles*, and finally in this folder the

documentation of the car *CAR1*. Since the car *CAR1* exists in different versions, such versions are collected in a *version cluster*. By selecting this cluster, a different view is presented to the user: this view shows different *configurations* of the car. The user selects the "automatic transmission" configuration and opens it. Again a different view is presented to the user. This time it shows the time evolution of the document - *editions*. When the user selects and opens one edition, the fourth kind of a view is opened: the *internal structure* of the document (please refer to the Figure 1. showing the internal structure of a document). The user now can browse in the chapter structure of the document. S/he sees the chapter title and some indications about the content of these chapters. For pictures their captions are displayed. Finally the picture is found.

The link information can be used in the following scenario: The user locates in folder *manuals / automobiles* the description of the car *CAR2*. By following the link, s/he may easily find the price information, that is generally located in a completely different folder.

Another possibility to locate objects in the document space is by specifying queries. This allows the location e.g. of *all documents of the class "internal memo" written by John between 1st August and 31st December*. After a query is executed, a user receives a list of objects satisfying this condition. Further investigations (e.g. by displaying each object) is used to locate the final object.

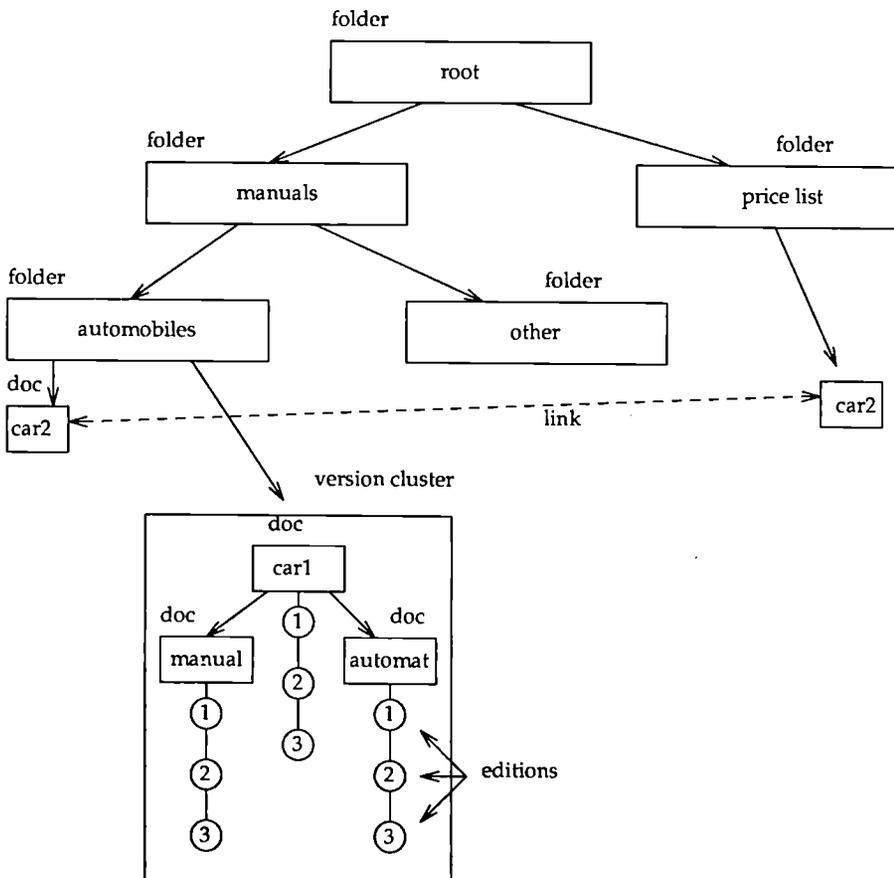


Figure 3. Example of document space

2.5 Integration into environment

As mentioned above, an important dimension of the creation of technical documentation is the fact that the sources for one document are distributed over many different physical "media" like paper drawings, paper photographs, external data bases, or external CAD systems, etc. It is considered essential that SPRITE can be smoothly integrated into such existing information environment. This integration is supported in the following dimensions:

- Technical drawings on paper with a size up to A0 can be scanned and converted into the SPRITE document format. With a help of a tool, a smaller part of such large drawing can be "cut" and included both as a raster graphic or as a recognized drawing into the technical documentation.
- Paper documents consisting of both text and graphic can be scanned and converted into the SPRITE document format as well. The recognition process does not recognize a plain content only, but an attempt is made to recognise the logical structure of the document as well.
- Data contained in external relational data bases can be accessed and transformed to SPRITE documents. The ORACLE relational data base running on a VAX/VMS was chosen as a representative example of external data bases. In a final product additional data bases could be added.
- Three important formats that are often used or generated by CAD systems can be translated to SPRITE documents: AutoCAD, Calcomp, and IGES. By providing access to those formats, a large number of CAD drawings can be included in SPRITE documents.

3. Evaluation

The above mentioned solutions to the handling of a technical documentation's specific properties are based on investigations of user needs, on personal experiences of all partners of the consortium, and on studies of comparable systems. In several areas Sprite proposes new, innovate solutions that have no parallel in products currently being used for technical documentation. As it is necessary to check the acceptance of such new concepts in order to know the user feedback, an additional activity was started in the last phase of the project - the evaluation of SPRITE.

Since the environment (hardware, software, official standards, de facto standards) has changed since the project was started (1989), it was first necessary to reinvestigate the current situation - the needs of users in the year 1991. Such needs are being correlated with the approach taken by SPRITE - to confirm how far the original assumptions are still valid, which functionality of SPRITE is needed in the daily life of a technical author, or which functionality is possibly not required.

For the functionality that is both required by users and provided by SPRITE, the evaluation proceeds to a further step: the SPRITE solutions will be checked for user interface and system performance acceptance.

When the final version of this report is written (end of July 1991), the methodology for evaluating Sprite and questionnaires for visits have been worked out. Further first user visits have been carried out. No final results are available now, but they will be ready for the presentation during the ESPRIT conference in November 1991.

4. Conclusions

The result of the SPRITE project is a working prototype system for technical documentation. This system supports the essential functionality needed for the every-day work of authors, managers, and readers. The prototype is implemented on Sun workstation using OpenLook as the standard for user interface. The system consists of the following components:

- Multimedia data base keeping all persistent data (based on the commercially available relational data base system Sybase).
- Desktop maintaining current working objects for each user.
- Browsing supporting locating of documents or parts of them.
- Document editor with an easy-to-use typographical support.
- High volume laser printer.
- Scanning and recognition tools for documents and drawings (up to A0).
- Data acquisition tools extracting data from CAD systems and external data bases.
- Optical disk storage used for archiving of documents.

The working SPRITE prototype allows all partners to verify the acceptance of the concepts developed during the project and to carry out further experiments by testing the use of technical documentation in practice.

The marketing investigation proved that there is a real need for technical documentation systems. The ongoing evaluation activity will bring more details about specific needs.

The main result of the project is that each of the contributing partners now has the enabling knowledge to develop products in the area of technical documentation. It is expected that product based on this knowledge will be soon introduced to the market.

5. Acknowledgement

The work described in this paper was carried out within the SPRITE project. SPRITE is the project 2001 of the ESPRIT program supported by the Commission of the European Communities. Ideas of this paper were developed by a joint effort of all partners of the project.

- The main components of the project were developed by the following partners:
- Océ Nederland B.V. (Netherlands) project management, document editor, printing, desktop
- Alcatel TITN (France) scanning and recognition of documents and for optical disk
- AEG Electrocom AG (FRG) scanning and recognition of drawings
- Economisch Instituut Tilburg (Netherlands) multimedia data base
- Trinity College Dublin (Ireland) retrieval, extraction of RDB data

6. References

- [1] Office Document Architecture (ODA) and interchange format, ISO standard 8613
- [2] System Services Overview, Sun Microsystems 1990
- [3] Christodoulakis, S. et al, 1986. Multimedia Document Presentation, Information Extraction, and Document Formation in MINOS: A Model and a System. ACM Trans. on Office Information Systems, 4(4).

- [4] Carey, M. et al, 1986. Object and File Management in the EXODUS Extensible Database system. Proc. Int. Conf on VLDB, Kyoto.
- [5] Hederman, L., Weigand H., 1990. Versioned Documents in a technical document management system; ESPRIT 90, Proceedings of the Annual ESPRIT Conference, Brussels, Kluwer Academic Publishers
- [6] Kent, W., 1989. An overview of the versioning problem. Panel Contribution. Proc. ACM SIGMOD.
- [7] Klahold, P., G. Schlageter, W. Wilkes, 1986. A general model for version management in databases. Proc. Int. Conf. on VLDB, Kyoto
- [8] Kronert, G., 1987. Standardized Interchange Formats for Documents. ESPRIT'87, North-Holland.
- [9] Tichy, W.F., 1985. RCS - A System for Version Control, Software - Practice & Experience 15, 7(July 1985),
- [10] Walter, M.E., 1988. Technical Documentation Systems. Wending Your Way Through the Land of WYSIWYG; Seybold Study of the Technical Documentation Market and Products, 1988

SECURITY LEVELS SUPPORTED BY THE COMANDOS SECURITY ARCHITECTURE

Manuel Medina & Ana Moreno
Universitat Politècnica de Catalunya
Dept. of Computer Architecture
c/ Gran Capità, s/n - Mod. D4
E - 80834 - Barcelona (Spain)
Tel.: +34-3-401 6984
Fax.: +34-3-401 7055

Summary

This paper describes the Security functions supported by the COMANDOS kernels. These functions are grouped in sets, and each of them is identified through an index, the security level. The security level (SL) is a number that reflects the set of security services to be used on a COMANDOS system, and the quality of service to be provided by each of them on each invocation.

This paper states the security services selectable by a COMANDOS security administrator, and its logical combinations to be supported by the system in the project framework. The specific procedures to be used for each security service, at each level, are installation dependent, and must be agreed COMANDOS-Domain wide.

The paper is structured in three parts:

- One introduction to clarify the basis on which the development decisions have been taken,
- The description of the security functions to be provided at each security level by each service, and
- Some considerations about the achievable security levels, according to the ITSEC standard promoted in Europe.

1. Introduction

The COMANDOS security architecture is based in the Distributed Office Architectural Model (ISO DIS 10031). The security services to be provided are those recommended in the Security addendum to the Open Systems Interconnection Reference Model (ISO 7498-2), in the data exchange aspects. The more general security aspects of the operating system have been selected according to the recommendations given in the Information Technology Security Evaluation Criteria (ITSEC), in order to allow further certification of the achieved security level, according to the European standards.

The security components and protocols have been selected according to ECMA recommendations about "security in Open Systems" (ECMA-138).

The most important aspect of the COMANDOS security architecture is its modularity, i.e. the capability to select different sets of security services at installation time. This allows to select the adequate services and mechanisms, depending on the Risk Evaluation results, which will consider the environment on which the system will run. To fulfill this requirement, each security service has been developed independently of the others, and has 3 kinds of agents or components:

Server agent, to perform the service. Only few of them are required in each COMANDOS security domain.

Client agent, to interact with the user in each comandos site, and forward its security service requests to the server agent.

Management agent, to provide restricted access to the server agent, in order to modify its functionality.

ITSEC is a document developed under the sponsorship of several European countries, as an input to produce an European standard to state the security functions to be requested from a computing system. It comprises two parallel streams of evaluation: Correctness and Functionality. The first one focuses mainly on the design and implementation process, whilst the functionality classes are based on the security functions provided by the system.

ITSEC proposes 7 correctness criteria, hierarchically ordered, which main characteristics, relevant to COMANDOS project, may be summarized as follows:

- E0 Inadequate.
- E1 Informal description of the architectural design, and functional testing.
- E2 Informal description of the detailed design, and configuration control.
- E3 Source code and hardware drawings of the security mechanisms, and evidence of its test.
- E4 Semiformal style of specification, and vulnerability analysis.
- E5 Close correspondence between the specification and the implementation.
- E6 Formal style specification.

In addition to them, 10 functionality classes are defined:

- F-C1, F-C2, F-B1, F-B2 and F-B3: the first 5 are hierarchically ordered, and correspond closely with the US Trusted Computer Security Evaluation Criteria (TCSEC).
- F-IN for systems with high INtegrity requirements.
- F-AV for systems with high AVailability requirements.
- F-DI focuses mainly in Data Integrity during data communications.
- F-DC focuses mainly in Data Confidentiality during data communications.
- F-DX for networks with high demands on confidentiality and integrity.

2. Security Services

The COMANDOS systems may support the four services herein described, with the indicated possible levels of security for each of them. The description of each of the services includes references to the corresponding functionalities required in the (ITSEC), to certify a given security functionality (F-nn) or security Assurance-Correctness level (En). Due to a lack of formal specification language usage in the design of the COMANDOS system, the upper limit of security assurance level is E3, and the affordable functionality in the project framework is up to F-B2 and F-DX, but functionality F-IN could easily be reached with a few further work.

2.1. Authentication

There are 3 possible authentication levels, depending on the kind of credentials used to authenticate the users of this service, and the way they are sent to the authentication server.

0: No credentials (useful in single user PCs or WorkStations).

1: Username/Password (in clear) (like in the current UNIX systems).

2: Username/Key (cyphered) (like in the Kerberos system).

All functionalities require at least level 1. Functionality F-C2 requires the system to be able to state the identity of the user at any time, and this means in practice the need store some kind of user credential, to avoid requesting it from the user each time the system requires it. In this case, the credential of the user must be stored encrypted, to fulfill any functionality.

2.2. Access Control

Each of the 4 supported levels corresponds to a stepwise refinement of the complexity of the authorization server to be used in the object invocations. Level 1 does not allow management of Access control lists (ACL) at run time, with independence of its number, because this is considered a first step of the authorization service, and the number of ACLs defined in the object are not relevant to the invoking object, since it will see only one in any case. Once introduced the ability to support ACL management operations, in levels 2 and 3, the difference between them depends on its ability to manage several ACLs.

0: No authorization control.

1: Single ACL consideration, no distinction of views of object, i.e. all users authorized to access an object may perform the same set of operations on it.

2: Operation based ACLs. Association of one ACL to each view of the object. This functionality allows to state specific rights to invocations coming from different users, i.e. roles are supported.

Level 1 would fit F-C1 requirements, whilst level 2 is required to fulfill F-B1 requirements.

2.3. Audit

In this section we describe in fact the requirements made for both audit and accountability purposes, according with the ITSEC document. From its point of view, only the capability to examine the accountability files is required from the Audit service. The proposed events to Audit correspond to a stepwise approach to the final requirement, increasing the granularity of the recorded events at each new level, i.e. logging more fine object access or management at each higher level number, which should logically include all the previous kinds of events.

For each kind of event, three possible set of events could be recorded: none, only fails, any attempt.

The kinds of events to record at each audit security level are:

0: No audit records.

1: Object creation and deletion, considering the word "object" in a wide sense, i.e. including human users representing objects, sites, and containers. This level covers the main system configuration changes, and probably the most relevant for security purposes. It is useful mainly for system management purposes, and also for accounting.

2: Extent (1) creation, deletion and modification, including addition and deletion of trusted classes from the set of those allowed to be downloaded in the extent.

3: Job and activity creation. This information may have accountability implications, since includes the creation of the job associated to each user that logs into the system. The activities created from a job, in addition, may help in the record of the amount of usage made of the system during the session.

- 4: Change of ownership and in general change of ACL of an extent. This action may have implications in the security of the system, since the extent is mainly a set of objects that will be able to interact among them without the control of the COMANDOS platform, since they trust each other, and change of its owner may allow further modifications on its composition.
- 5: Change of ownership and in general change of ACL of an objects.
- 6: Access to nodes, i.e. establishment of session between two nodes.
- 7: Cross context object invocation. The record of this kind of actions is required by the ITSEC document if it is either "... subject to the administration of rights" or "... affects the security of the system". This means that invocations to some critical objects like ACLs, must be logged, but that invocations between trusted objects, belonging to the same extent, do not need to be logged in the audit files. The record of any object invocation in the audit files may be useful for debugging, and recovery purposes mainly, but also provides the finest information needed to analyze security failures.
- 8: Secure data exchange over the network. This includes in fact a selective audit of the data being transferred between two extents located in different sites, which includes object migration.

The ITSEC book specifies the required data for each of the logged type of action in the system, and, details apart, should include: Date; Time; user identity; Object(s) identity; Type of action (attempt); and success or failure of the attempt.

Functionality F-C1 does not require any of the levels just explained. Functionality F-C2 requires only level 1. Functionality F-B1 requires levels 1 to 72, according to the ITSEC. Well, in fact the things are not so clear, because some readings of the ITSEC document would also allow to get F-B1 certification just with levels 1 to 3, i.e. without recording object migrations and object invocations, if these actions are considered not relevant for accountability or security purposes, which is easy to justify if we consider the actions already logged at levels 1 to 3. In particular, if the system does not allow changes in the ACLs while the object is in memory, i.e. they must be changed from the class definition, the record of object invocations should not be considered mandatory to reach functionality F-C1. To finish, functionality F-B2 requires all the audit security levels herein described.

2.4. Data Exchange

This last component of the security level corresponds to the lower end of the security functions. Its implementation may depend on other COMANDOS subsystems, like the Transaction or the communications, or even be transparent to the COMANDOS platform, and be directly supported by the native O.S. or micro-kernel. The 4 levels considered for this service correspond to the OSIRM standard (ISO - 7498 DAD2), and to the quality of services recommended for its transport layer.

- 0: Clear transmission/storage, i.e. not encrypted. Does not provide any protection to the information.
- 1: Integrity. The adequate mechanisms needed to guarantee the integrity of data are used. Examples are: Sequence numbers, digital signature, checksum, etc. The specific mechanisms to be used to provide that level of security in the data interchange are installation dependent, and have to be agreed at domain level. This level is specifically required in ITSEC functionality F-DI.
- 2: Confidentiality. This quality of service implies the encrypting of the whole data interchanged, or even stored in the system. This level is specifically required in ITSEC functionality F-DC.

3: Both mechanisms to be applied simultaneously (integrity and confidentiality). This level is specifically required in ITSEC functionality FDX.

Only level 1 is required to reach the F-C2 ITSEC level of functionality, in order to "allow the system to state the identity of the user in every interaction", through the signature of the messages by the originator. Levels 2 and 3 of data exchange service, as just described here are only required for functionality FB2 of the ITSEC document, but are kept here for information, to allow further extensions of the system, and because the existing implementations of secure communication systems consider the four choices, to be fully ISO conforming.

3. Distinguished security levels

It is evident that not all the combinations of the levels of security offered by each of the services are useful. This has two consequences:

- First is that the encoding will already merge a few of the combinations, to reduce the number of possible combinations to a suitable figure.
- Secondly we will recommend several security levels, suitable to fit an ITSEC functional requirement, or useful for the user.

Probably it would be interesting to select some of the possible combinations of services, to deal with two objectives: to avoid the illogical combinations, and to help the user to select the more useful ones, in order to avoid unbalanced usage of each of the security services, that would result in a waste of resources without increase of security.

At this point we have to consider whether the security level is used with invocation or system scope. In the case of a unique system wide security level (SL), it is not useful to limit its set of possible values, because the system security officer or system administrator will have enough requirements to fix the needed QoS (value) for each of the services, according with considerations like these:

- Audit functionality will be chosen according with the constraint between accountability granularity and system overhead.
- Data Exchange will be chosen according with the estimated probability of intrusion in the network.
- Access control functionality will be selected also according to the intrusion probability, but will also be restricted by the capabilities of the language used to specify the objects, since the existence of several views and consequent ACLs is dependent of the capabilities offered by the programming language used to specify that.

All this makes very difficult to select a subset of SL values with any general interest. What may be interesting for the system security officer, is the minimum level of security of each service, required to get a given ITSEC functionality certification.

4. Conclusions

According with the considerations made up to now, we can conclude two things:

- A fixed security level could be enough to cover most of the security requirements in systems running fixed applications, always with similar security requirements.
- For general purpose systems, open to any application, and with a variety of user requirements, some kind of security service level selection has to be offered to the security officer, in order to allow him/her to trimmer the system and adjust the functionality to the known risks. In this case, it would be enough to check the required security services when "something" is created, where "something" may stand for node, job or object.

To allow the user to select the security level at object creation or invocation, it should be possible to specify which services have to be used to:

- check the authenticity and access control between the creator and the created objects, and also
- to specify the characteristics of the channel that will carry the messages between them (audit and data exchange choices).

.. This could be done through the different allowed views of an object, if the Access Control service supports them through the ACLs. If the user chooses the security level at job creation, means that the same security services will be used during the whole interactions between objects in that job, i.e. there will be one security level associated to each active user on the system.

References

(ISO7498), ISO Open Systems Interconnection Reference Model. Addendum 2 on security.

(ECMA-138), Security in Open System: Data elements and service definitions.

(ITSEC), Information Technology Security Evaluation Criteria. Harmonized criteria. Ver. 1.1. 10 Jan 1991.

(GUIDE), COMANDOS architecture.

(ISO-10031), Distributed Office Architectural Model.

(1) Extent is the set of objects in virtual memory, that trust each other.

SUPPORTING OBJECT ORIENTED LANGUAGES ON THE COMANDOS PLATFORM

Vinny Cahill¹, Chris Horn, Gradimir Starovic
Distributed Systems Group, Dept. of Computer Science,
Trinity College, Dublin, Ireland.

Rodger Lea
Chorus Systèmes, 6 av. Gustave Eiffel,
78182 St. Quentin-en-Yvelines, France

Pedro Sousa
INESC, Rua Alves Redol, 9, 1000, Lisboa, Portugal

Abstract

The Comandos project is designing and implementing a platform to support distributed persistent applications. In particular the platform supports the object oriented style of programming. An essential requirement of the Comandos platform is that it must support applications written in a variety of existing as well as new (object oriented) programming languages. Moreover, the platform must support interworking between different languages. Each language may naturally have its own object model and execution structures implemented by a *language specific* runtime system. Rather than forcing each language to adopt a common object model and execution structures in order to exploit the distribution and persistence support provided by the Comandos platform, Comandos provides a *generic runtime* system on top of which individual language's specific runtimes may be implemented. In this paper we show how a language specific runtime for an existing language such as C++ can be constructed above the Comandos generic runtime.

The growing use of distributed systems reflects both technological and organisational evolutions. Advances in computing and networking have led to the use of local-area distributed systems composed of heterogeneous workstations and servers. Human organisations are often, by their nature, distributed, but with strong requirements for interworking and overall integration within the enterprise. This evolution leads currently to the concept of "collective computing" [3], in which the overall application is constructed or assembled from many components, each performing its own specialised function.

The development and integration of application software is currently a labour and cost-intensive proposition, particularly for distributed applications which handle large volumes of structured data. Methodologies and tools are needed to master the complexity inherent in heterogeneous distributed environments and application requirements.

Comandos is primarily targeted at the development and support of integrated distributed applications within a *cell*, which constitutes the basic organisational and administrative component within an enterprise. A cell is composed of a set of cooper-

¹ email vjcahill@cs.tcd.ie

ating workstations, servers and processor pools connected through a high-speed local area network. The goal is to present the distributed system as a coherent entity to its users despite the variety of its components, and to provide the enabling technology for reducing the cost of distributed application development and maintenance.

The overall objective of the Comandos project is to identify and construct an integrated multi-language application support environment for developing and administering distributed applications which can manipulate persistent - long-lived - data. It is intended to provide such an environment in the framework of multi-vendor distributed systems. This platform will allow both the development of new applications, and the co-existence with old-style (Unix² oriented) applications.

The Comandos platform includes infrastructure for:

- distributed concurrent computations;
- storage and retrieval of persistent data;
- multiple programming languages;
- reusable and extensible software modules;
- secure and protected data;
- on-line management, monitoring and control;
- access to pre-existing applications and information systems;
- interworking with non-Comandos environments.

The project is innovative in that it is integrating operating systems, programming languages and databases technologies. The unifying view is provided by a model and system architecture based on the object-oriented approach, coupled with persistent distributed storage.

Within the comandos project, the object paradigm provides the basis for an integrated view of application construction, and system management and control.

The project started by defining a conceptual model for structuring distributed applications. This model defines the comandos virtual machine, and provides an object-oriented view of the distributed environment.

Based on this model, the project is providing a multi-lingual environment which supports the concepts of the model through various languages (initially C + + [5], Eiffel [4] and the Comandos object-oriented language, known as Guide [2]).

In this paper we describe the support provided by the Comandos platform for language implementers and show how an existing language can be adapted to use the facilities of the Comandos platform.

1. The Comandos Virtual Machine

The Comandos Virtual Machine is internally composed of six components:

- the Execution Subsystem (ES) provides support for object execution and the Comandos notion of a distributed process;
- the Virtual Object Memory (VOM) handles all operations related to the manipulation of virtual address spaces;
- the Storage Subsystem (SS) provides long-term storage for persistent objects;
- the Transaction Subsystem (TS) supports the execution of atomic transactions;

² Unix is a trademark of Unix Systems Laboratories, Inc.

- the Communication Subsystem (CS) is responsible for providing a generic RPC interface independent of the underlying stack of protocols;
- the Protection Subsystem (PS) ensures the specified level of protection during application execution.

The implementation of these components is split across three interfaces.

- the *Virtual Machine Interface* - The interface across which a supported language communicates with the Comandos platform;
- the *Kernel Interface* - Dividing those parts of the platform which are accessible directly by applications ("user" mode) and those which are accessible only in a privileged mode (i.e. the Comandos kernel);
- the *Environment Interface* - The interface from a Comandos implementation to its underlying hosting environment: Unix or micro-kernels like CHORUS³.

The Virtual Machine Interface is the uniform view presented by the Comandos platform to each of the various supported languages. It is provided by the primitives of the *Generic Runtime* (GRT) layer that itself interfaces with the services of the underlying Comandos kernel.

The GRT implements the local object space directly manipulated by application programs i.e. it is mainly concerned with local object management. The GRT is implemented by code running in user mode within each address space. Services such as address space creation and deletion; remote invocation; sharing of objects between address spaces as necessary and object location within the distributed system may be implemented in kernel mode or by specific servers as well as in user mode within and address space, depending on the underlying environment.

As different languages have different calling semantics, a *language specific runtime* must adapt the GRT primitives to the language specific format. Moreover, as most of these primitives are based on manipulation of objects, whose format and model differ in each of the different languages, each specific runtime must also hide these language dependencies from the GRT support.

To provide this flexibility and to make a minimum number of impositions on any language, we propose a general model in which the language makes calls to the GRT but expects the GRT to understand little of the semantics of these calls. To deal with the problem of language specific information, the architecture makes heavy use of *up-calls*, where an up-call is a call from a lower level to a higher one, using an entry point previously supplied by a regular call (*down-call*).

This two-way interface between a language-specific runtime and the GRT allows objects of heterogeneous languages to be handled commonly by the GRT. This scheme is sufficiently generic so as to allow flexible implementations of both the generic and language specific runtimes.

The next section gives an overview of the basic functionality of the GRT.

2. The generic runtime

The GRT provides fine grained passive data objects (which will be referred to, in this and subsequent sections, simply as *objects*) to which a language, through its language specific runtime, may bind class code. Language level object models can thus be built.

³ CHORUS is a trademark of Chorus Systèmes

An object may be uniquely identified by its so-called *global name* which is sufficient to locate the object even when it is mapped on a remote node or stored in the storage subsystem.

The GRT provides generic support for local management of these passive objects, including their creation and possible garbage collection. Objects, which are viewed by the GRT simply as contiguous blocks of memory, are used to contain and enclose language level objects (referred to as *language objects*) containing both direct data and references to other language objects.

Supported languages (i.e. their language specific runtimes) can build language specific forms of object references (referred to as *language names*), such as tagged pointers, which may be stored in language objects, and thus in objects while they are mapped into some address space.

In particular, the GRT supports direct addressing over the objects mapped into an address space. The identifiers for objects communicated between the GRT and a supported language take the form of direct addresses⁴.

The protocol between the GRT and the language (specific runtime) - the upcall mechanism - allows the GRT to perform a number of operations on language objects, such as location of language names stored in local objects and conversion of these names to/from global names as objects are mapped and unmapped. This protocol requires that each object must have a minimum set of methods, one for each basic operation which the GRT may request.

Object Creation and Promotion

Objects are brought into existence by the Create call. Create allocates space, which is untyped, and will be used by the language specific run time as a repository for language objects.

Any object created in this way is known as a *volatile*. Such objects exist and are known only within the address space in which they were created. Objects may become known outside of their address space of creation (either because a reference to the object has been passed out of the address space, or because the object itself migrates out of the address space). If this happens, the object is said to have been "promoted" to being a globally known object.

Actually achieving this promotion is a two stage process. First, the object is registered with the GRT by the Register call which associates an upcall table to the object allowing it to be managed by the GRT.

To become globally known, the language specific runtime must call the AssociateGName function which returns a global name for the object.

The object cannot now be garbage collected using address space local information alone. The object may be unmapped on address space deletion or when no longer required by the language run-time.

Object Global Naming

Objects are assigned global names (only) when they are promoted. Thus a location independent form of naming is used to transmit object references outside of the address space in which the object was created. Global names may be passed in remote/cross-

⁴ Note that a language name itself is not necessarily a direct address.

address-space invocations and stored subsystem along with the object which contains the name.

The GRT interacts with the storage subsystem in allocating global names since a global name consists of a secondary storage container identifier and a generation number which is unique within that container. The container identifier names the secondary storage container initially specified for the object. If the object migrates between containers, then the initial container must track the object's current storage location. Volatile forwarding information may be used to accelerate a search, but may be out of date or incomplete due to node crashes.

Object Invocation

Object invocation is the basic primitive of the model reflecting all the features related with transparent handling of persistence, distribution and sharing.

Invocations on objects mapped locally are performed at language specific run time level. However, attempts to access objects which are not mapped in the current address space - *object faults* - are trapped by the specific runtime which calls the GRT. It in turn, either arranges to map the object into the current address space, or arranges to carry-out a cross-address-space or remote invocation as required. The GRT may interact with the protection subsystem, storage subsystem and global location service in determining how to handle the object fault. The following paragraphs introduce the basic mechanisms within the GRT to handle object faults.

Object Mapping

When mapping an object, the GRT must first allocate memory for the object. The GRT translates global names contained in the mapped object to language names, by interacting with the mapped object's specific runtime. This translation may be delayed until the object is actually used (e.g. being invoked) if the object was pre-fetched. If the referenced object does not currently reside in the local address space, then the GRT may be used to build a local data object which represents the referenced absent object.

The translation from global name to language name is maintained in the GRT's Context Object Table (COT) within the address space where the object is mapped.

Groups of objects, i.e. clusters, form the actual unit of transfer between the GRT and the storage subsystem. Since objects are retrieved in clusters, the GRT must handle demand loading of one object and pre-fetching of the others. Pre-fetched objects are simply registered in the faulting address space's COT.

Remote and Cross-Address-Space Invocation

The GRT co-operates with the requesting language in performing remote and cross-address-space invocations. The GRT supports the building of parameter frames for transmission between heterogeneous machines by supplying - to language dependent RPC stubs (which may be generated by standard stub compilers) - generic routines to encode parameter data in network format. The language specific runtime can then perform marshalling into buffers provided by the GRT using the appropriate GRT routine to encode the data.

The GRT interacts with the communications subsystem and the protection subsystem to supply the mapping between the target objects global name and the authenticated remote invocation transport path.

Local Garbage Collection

The creation of new objects may lead to exhaustion of memory. To recycle unused memory, the GRT incorporates a local (per address space) garbage collector, taking externally known data objects (which have been assigned a global name) as its root, and which collects unreachable volatile objects.

.. As the GRT imposes no parameter frame formats, the garbage collector must be conservative when scanning stacks⁵. The local collector must cooperate with on-line global garbage collection, if it is present. Local garbage collection uses the upcall mechanism to locate object references held by typed language level objects. The garbage collection should be compacting only if dynamic object movement is supported by the language and its specific runtime.

3. Supporting an Object Oriented Language above the GRT

In this section we examine what the major technical problems are when mapping a language such as C++ onto the Comandos Virtual Machine.

Our goal is to transparently provide the C++ programmer with a distributed persistent programming environment. However, we are concerned to minimise the side-effects of using our platform for programmers. This generates two, conflicting, constraints: that we maintain, as far as is possible the current model and syntax of the C++ language, yet, we do not penalise C++ programmers who make no use of the enhanced functionality provided.

Both distribution, and persistence impose a common requirement on the C++ programming model for which current compilers do not cater. All references to objects may actually point to objects that are not mapped in the current address space i.e. objects which are either mapped on a machine elsewhere in the system, or are currently inactive and so are on secondary storage.

Since current implementations of C++ use virtual memory pointers to refer to objects we are faced with a dilemma: either we adapt current compilers to generate globally unique, long lived object pointers, or we add support for the GRT to map the compiler generated local object identifiers to global ones i.e. those generated and managed by the Comandos GRT.

Another problem is the need to deal with pointers to context dependent data, for example file descriptors. If we pass a file descriptor out of an address space, without some means of conversion, it may be impossible to use that descriptor in any other address space.

In summary, the language designer has three main problems to solve:

- Dealing with context dependent data.
- Managing pointers to stored objects.
- Managing pointers to remote objects.

3.1 Context Dependent Data

We adopt an approach to context dependent data that requires programmer intervention. Our motivation is that such data is outside the scope of the object oriented

⁵ Stack allocated objects are never visible to the GRT.

world, and, more importantly, the programmer making no use of distribution or persistence should not be concerned with the problem.

The approach is to encapsulate such data within a type that is capable of dealing with distribution or persistence.

Consider for example a file pointer; if we create a function that the system can call whenever a file pointer is passed into or out of an address space, to open or close the correct file, then we can correctly support this pointer.

3.2 Pointers to Stored Objects

Since an object may contain references to objects that are currently in the persistent store, we must ensure that any attempt to access such an object causes an 'object fault'. We are currently experimenting with two mechanisms for this purpose: The first mechanism involves the use of a proxy object to represent the stored object such that when accessed, the proxy is responsible for locating the correct object and overlaying itself. The alternative mechanism is to have references to absent objects point to invalid memory so that attempts to access such objects cause memory faults. Thus we can trap the 'object fault' without using a proxy and can load the required object in the fault handler.

3.3 Pointers to Remote Objects

The solution to the last major problem, that of trapping invocations to remote objects and forwarding the request uses a model similar to that described above. However, in the remote case we are always constrained to use a proxy so that accesses to the data of the object can be caught. Using a proxy object in place of the remote object, we catch invocations, locate the remote object, marshall the arguments and send the remote request using the support provided by the GRT.

4. Using the Comandos GRT

To illustrate how the Comandos GRT supports the C++ language, we detail the creation of a C++ object on the Comandos platform and the subsequent invocation of another C++ object.

4.1 Object Creation

In traditional languages such as C++, language objects can be instantiated in several possible scopes. An object can be declared in a global scope (outside of any function or class), and thus reside in the data space of the application; can be dynamically allocated in the heap of the application, by means of mechanisms similar to `malloc` in Unix; can reside in the stack if declared within the scope of a function definition; and can exist within another object (wherever it may be).

However, this scenario of object creation is independent of the GRT, which does not require language objects to be located in a specific region. Although the GRT provides the `Create` call for allocating new space,⁶ we are never constrained to use only this space. The only requirement is that all *promoted* objects reside in memory known to the GRT. Thus not all language objects need be known to the GRT.

⁶ as described in section 2

Objects to which references are passed outside of the address space in which they were created (as described in section 2) need to become known to the GRT. This promotion is an operation carried out on demand. In fact, whenever references are sent outside of the address space, a test needs to be made to determine whether or not they refer to globally known objects. If they refer to volatile objects, then these objects must be promoted by calling the Register and AssociateGName primitives as described previously. The test is carried out by language specific code, i.e. as part of the upcall to the appropriate object.

When the Register operation is called, the language specific run-time must provide the binding to the correct upcall functions to allow the GRT to manage the newly promoted object with respect to persistence, distribution and context dependant data.

This code, that encapsulates context dependant data and the upcall functions that support persistence,⁷ may be generated automatically using either pre-processor, or compiler adaptations, or may be hand generated by the programmer.

The reader is directed to appendix A for a full description of the upcall functions that must exist for each GRT managed object.

4.2 Object Invocation

When an invocation is attempted using a virtual memory pointer, a number of events may be triggered depending on the state of the corresponding object:

- The object is known locally and the invocation proceeds without use of the GRT.
- The object is unknown locally, causing an 'object fault', passing into the GRT and asking the GRT to locate the object and map it locally.
- As above, however, the object is already mapped elsewhere or must be mapped elsewhere⁸, requiring a remote invocation to be made.

Here we are only concerned with the last two cases, as the first will be confined to the language specific runtime and makes no use of the GRT.

4.2.1 Map the Object Locally

When the use of a reference causes an object fault, control is passed to the GRT via the Invoke primitive passing as parameters the global name of the object and ActivationData - a structure holding the method identifier and parameters for the invocation. It should be noted here that the code handling the object fault must make a mapping from the local C++ virtual memory pointer to a valid Comandos global name.

Invoke causes the GRT to locate the object. We will assume here that the object will be mapped into the calling object's address space.

The GRT uses an internal service to locate the object. It then maps the object into the current address space (c.f. section 2). Once mapped, the GRT upcalls the object's GetArrayRef() function. This function returns the address of an array of references

⁷ as well as the proxy code to be used for remote objects

⁸ for example, because of heterogeneity

contained within the object. This array, in conjunction with the `GetVMemAdd()` upcall allows the GRT to obtain a list of pointers within the object to other objects.

For each reference within the object the GRT has a choice, either it can go ahead and load the referenced object, updating the pointer⁹ or it can arrange that the use of the reference causes an object fault¹⁰.

Once loaded, an upcall is made to the function `AfterMap()` which is responsible for ensuring that any context dependent data, such as file descriptors as described above are dealt with (3.1).

Once this has been carried out, the GRT can then assume that the object is correctly mapped, and initialised, and can upcall the function `Dispatch()`, passing it the `ActivationData` structure obtained in the `Invoke` downcall which allows the language level to call the correct method, after it has carried out any necessary work connected with stack management.

4.2.2 Map the Object Remotely

The scenario for dealing with an object that is mapped remotely is similar to that of a local invocation in terms of dealing with the mapping, converting global references etc. However, the difference is in the propagation of the invocation frame.

When the object is mapped remotely, the GRT transfer processing from the current address space to the remote address space¹¹. In this case the `ActivationData` must be transferred to the remote address space.

The main complication arises when we have to deal with heterogeneous architectures. In this case both the object itself (if it is being mapped from the storage subsystem) and the invocation data may need to be converted to the appropriate format.

As far as the actual object is concerned, in addition to the mapping and series of associated upcalls described in section (4.2.1), we may need to upcall `ConvertFormat(from, to)` which converts to or from a standard architecture independent format. In the case of mapping the object, it will be converted from a standard representation on disk to the required representation for the node.

Dealing with architectural differences at the level of the invocation data is carried out in a similar manner. The GRT, when it recognises that it is sending to a different architecture ensures that the outgoing invocation frame is converted to an architecture independent one using the `ConvertActivationDate(activationData, from, to)` upcall. On receipt of an incoming invocation, the standard representation is converted into one suitable for the receiving machine. Invocation then proceeds as before using the `Dispatch` function.

5. Conclusions

In this paper we presented the mechanisms provided by the Comandos platform to support a variety of languages in the comandos distributed environment.

We are currently in the process of implementing the Comandos kernel and GRT on Unix, CHORUS and Mach 3.0. Language specific runtimes for C + +, Eiffel and Guide are also being implemented. A prototype implementation, known as Amadeus, which

⁹ and recursively load other referenced objects.

¹⁰ Again via virtual memory techniques or using proxy objects.

¹¹ A lightweight invocation may occur if the object is in another address space but on the same side.


```

virtual VMemAdd      ConvertActivationData(ActivationData *actData,
                                           Architecture *from,
                                           Architecture *to, Error *error);
virtual String       ExplainError(Error *error)
virtual void         DisplaySelf()
};

```

Object defines the set of methods that all Comandos objects must support to be correctly managed by the system. Typically, these methods are generated automatically or are inherited by each object in the system.

AfterMap is up-called immediately after the mapping of the object into an address space. This allows the programmer to update any address space dependent data within the object.

BeforeUnMap is up-called immediately before unmapping the object from an address space. It is used for the same purposes as the AfterMap method.

Trap is up-called when a hardware or system exception is raised, and allows the language run-time to handle exceptions caught by the virtual machine.

Dispatch is up-called by the Generic Run-Time when an incoming cross-address-space invocation for the object is received, and when the object is mapped locally.

GetClass returns the global name of class object for this object.

GetVMemAdd retrieves the address of the object instance data.

GetArrayRef is up-called when the generic run-time needs to know the location of references within an object. The structure returned is an array containing header information and a list of offsets, containing the location of references within the object. The final address of the references is obtained by adding these offsets to the object location, returned by the GetVMemAdd up-call.

ConvertFormat is up-called when an object's representation need to be converted to one suitable for another architecture. Conversion between heterogeneous architectures is performed in two steps: from current format to a standard one (within the calling address space), and then from the standard to the desired format (within the called address space). Thus, one of the Architecture arguments is guaranteed to be a standard one.

ConvertActivationData is up-called to convert the parameters contained within an ActivationData structure. The semantics of this primitive is similar to the previous one.

B. Class Context

An instance of the Context class is created and properly initialised by the GRT when an address space is created. The operations available on this class describe the downcalls available to language specific runtimes.

```

class Context {
public:
    VMemAdd      Create(int size, Error *error);
    void         Register(VMemAdd vMemAdd, GName gName, Object *object,
                        Error *error);
    GName        AssociateGName(VMemAdd vMemAdd, Error *error);
    void         Invoke(GName gName, ActivationData *actData, Error *error);
    Bool         IsLocal(GName gName, Error *error);
};

```

```

void          LoadObject(GName gName, Error *error);
void          ReleaseObject(GName gName, Error *error);
void          SetCurrentObject(Object *object);
void          Group(GName *gName, U32 nbGname, Error *error);
virtual String ExplainError(Error *error)
virtual void  DisplaySelf()
};

```

An instance of the **Context** class is created and initialised by the GRT when needed.

Create is called to reserve a memory region for holding objects. Typically, languages use this primitive to allocate space for their objects. The Create primitive returns the address of the allocated memory.

Register is called to make the specified object (represented by the VMemAdd parameter) known to the generic run-time. The GName parameter indicates the global name of the class for this instance. The Object parameter contains the set of up-calls required for in the registration procedure, and must be previously created and initialised by the corresponding language specific run time.

AssociateGName() associates a global name to the object located at VMemAdd address. The global name is returned.

Invoke causes invocation to be transferred to the object specified by GName with the parameters specified by ActivationData.

IsLocal returns true if the object specified by GName is mapped in the current address space.

LoadObject maps the designated object in virtual memory so that it can be scanned without being invoked. The object stays locked in memory until it is explicitly released with the ReleaseObject primitive.

ReleaseObject informs the system the designated object is no longer needed in memory, and therefore can be stored back to disk.

SetCurrentSignalObject informs the system that the designated object should become the default handler for hardware traps occurring within the current address space.

Group is called to inform the GRT that the list of objects should be stored together. This call is advisory.

High fidelity, programmable CCD colour camera for desk top publishing and the graphic arts

H. Brunner, K. Engelhardt, M.T. Gale, G.K. Lang,
P. Metzler, J.M. Raynor and P. Seitz
Paul Scherrer Institute Zürich,
Badenerstrasse 569, CH-8048 Zürich, Switzerland

L. Brissot
Thomson TMS,
Avenue de Rocheplaine, BP 123, F-38521 St. Egrève, France

G. Cilia, C. Gadda and V. Leone
OPTEC s.r.l.,
Via Canova 10, I-20017 Rho (Mi), Italy

SUMMARY

A prototype CCD colour camera has been developed for still photography applications in desk top publishing and the graphic arts. The camera uses a single megapixel CCD imager with a unique colour filter mask and image microscanner module to achieve programmable resolution of up to 3k x 3k colour pixels. The use of multilayer dielectric filters results in very high fidelity colorimetry. A specially designed lens accommodates a wide range of image formats and houses the microscanner module. Fully digital colour processor electronics are used to produce the colour image data (RGB, XYZ, etc.) with high signal/noise, excellent reproducibility and high colorimetric accuracy.

1. Introduction

This paper describes a prototype colour CCD camera developed as part of the ESPRIT II project MASCOT (Multi-environment advanced system for colour treatment). The camera constitutes the image acquisition front-end of the MASCOT system, which itself consists of a high resolution image input module (the camera), a workstation and software for compressing, storing and processing the colour images, and a colour printer for reproducing the processed images. The intended applications and market potential for this system are in advanced colour desk top publishing (DTP) and in electronic photography for professional usage such as in the graphic arts; a typical example would be the acquisition and processing of images for creating a high quality colour catalogue.

The MASCOT camera has been designed to incorporate a number of advanced and unique features which are not available in cameras which are currently commercially available. The camera uses a single CCD imager with a unique aperture and colour filter mask to achieve very high resolution and colorimetric performance. The major features, shown in Figure 1 and described in the following sections in more detail, can be summarized as follows:

- **Wide range of input formats**

A custom designed lens for the 20x20 mm² imager has been optimized for DTP applications. The lens accommodates object input formats from A7 to A0 (at working distances of 94 mm to 1450 mm), as well as objects at infinity.

- **Programmable, high resolution**

A 1024 x 1024 pixel CCD imager format with a novel mask aperture array is combined with the ability to translate the image in sub-pixel increments ('microscanning') using a modified lens assembly. This results in a programmable resolution of up to 3072 x 3072 full colour pixels, corresponding to very high quality images.

- **High quality colorimetry**

The use of specially designed and fabricated multilayer (dielectric) colour filters enables the optimum filter characteristics determined by computer simulation for the complete camera system to be realized. The result is a colorimetric performance better than commercial studio television cameras, and suitable not only for the acquisition of colour images, but also for high accuracy colour measurement tasks.

- **Fully digital colour processor**

The video output from the CCD imager signal processor is digitized (pixel synchronous) with 10-bit accuracy. All subsequent processing of the signal (colour demultiplexing, matrixing, etc.) is carried out digitally, resulting in excellent signal/noise and reproducibility in the acquired images, as well as the ability to precisely choose the processing parameters to match different illumination conditions and output devices.

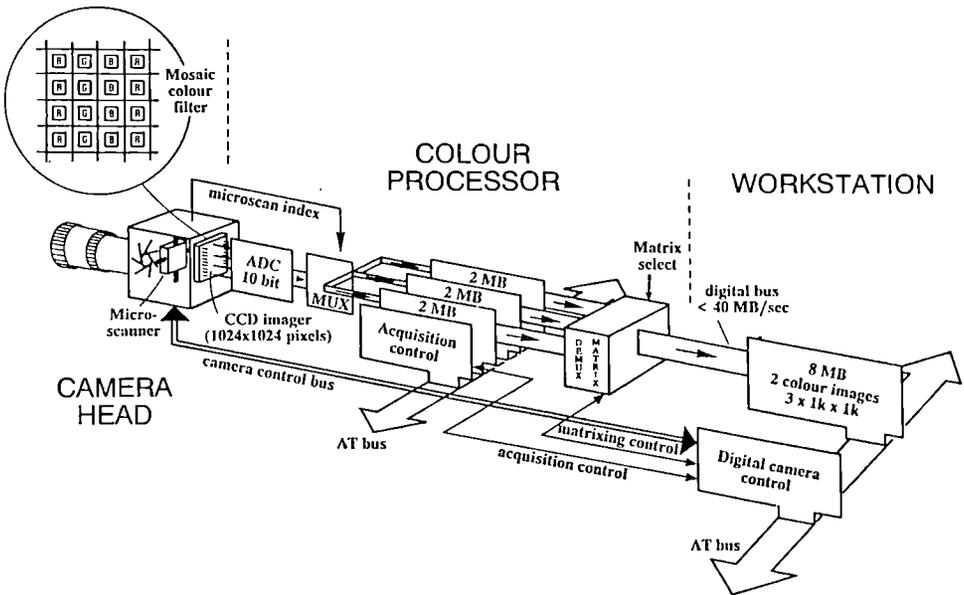


Figure 1. Overview of MASCOT camera system

The MASCOT prototype camera represents a significant improvement in flexibility and performance over existing colour cameras. Microscanning techniques for increasing camera resolution have been described in the past by Lenz []; in the MASCOT camera the individual pixel apertures were optimized for the microscanning approach and combined with specially designed colour filters and fully digital signal processing

to achieve a very high performance colour camera. As an input device for graphic arts and DTP applications, the MASCOT camera is particularly suited to the acquisition of images of 3-dimensional objects with high resolution and accurate colorimetry. The application areas include:

– **Desk top publishing**

Acquisition of images of 3D objects (not possible with flat bed scanners) and 2D objects for DTP, catalogues, publications, data banks etc. Programmable high resolution and good colorimetry form the basis of a flexible, high performance system.

– **Works of Art and Architecture**

Image acquisition of paintings, sculptures, buildings etc. for documentation, archiving, restoration studies etc. High resolution and exact colorimetry are essential requirements.

– **Medical images**

Very high resolution microscanning and large dynamic range for digitizing radiological images.

– **Electronic still photography**

High quality image acquisition for storage or transmission over data links.

– **Colorimetry**

Direct measurement of colorimetric data for 3D and 2D objects. High signal/noise, repeatable colorimetry measurements of up to 10 million points per image. Digital controls to accommodate various illumination conditions and output devices occurring in practice.

2. Camera system overview

Figure 1 shows an overview of the basic components comprising the MASCOT camera system - their features are summarized here in order to explain the working of the camera system and are described in more detail in subsequent sections.

Lens

The lens (see Section 3) has been designed to satisfy the requirements of advanced DTP systems, namely a wide range of input formats (A7 to A0), high resolution (corresponding to 3k x 3k pixels over the 20 x 20 mm² image area) and good colorimetric performance. A microscan module (realized as a motorized tilting plate) is built into the lens housing and enables the image to be translated with sub-pixel (xy) accuracy with respect to the imager; this feature enables the single chip MASCOT camera to achieve the performance of a very high resolution 3-chip camera (see below).

CCD colour imager

The basic imager is a megapixel CCD imager (see Section 4) with 1024x1024 (1k x 1k) square pixels. A stripe colour filter (see Figure 2a) is mounted over the imager and serves two functions.

Firstly, the colour filter stripes aligned to the pixel columns encode the colour information in triplet stripes; the filter transmission characteristics of the 3 filter types (KLM) together with the infra-red (IR) filter in the lens were optimized by computer simulation to produce high colorimetric quality in the demultiplexed the RGB output of the camera. Filter fabrication as multilayer dielectric stacks and design by advanced

simulated annealing techniques enabled the desired transmission characteristics to be realized in practice, resulting in colorimetric performance difficult to achieve using conventional dye filters.

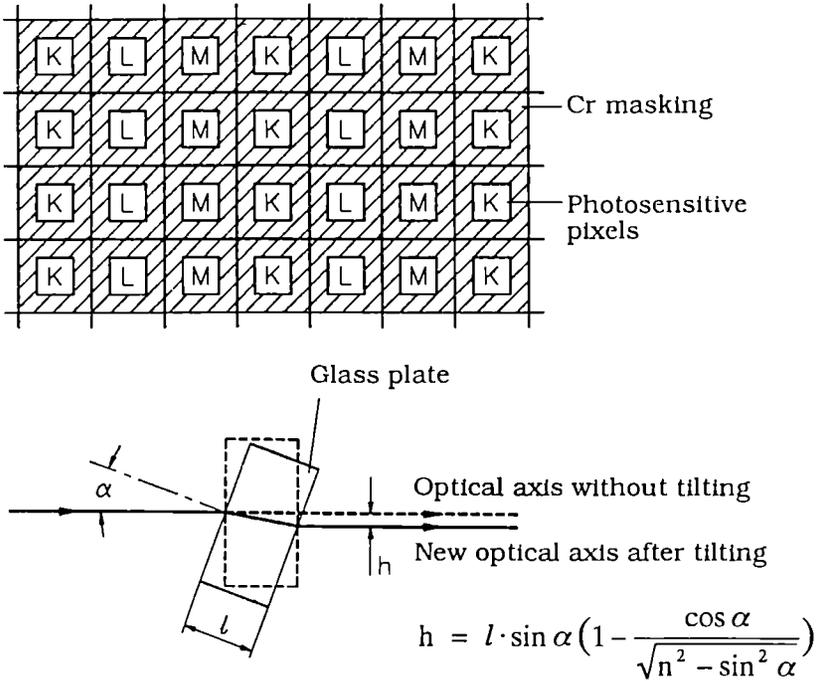


Figure 2. (a) Colour filter (b) Microscanning technique

The second function of the filter is to limit the aperture of each pixel to enable high resolution to be achieved by microscanning techniques. The basic imager with vertical triplet filter colour encoding can produce a colour image with 340H x 1024V full colour (RGB) pixels, which is totally inadequate for high quality DTP. In order to increase the resolution capability of the camera, the microscanning module built into the lens as a tiltable glass plate (see Figure 2b) enables the whole image to be programmably translated with subpixel resolution by up to 3 pixels. Acquisition of 3 images with 1 pixel horizontal translation between exposures enables a full 1k x 1k RGB image to be built up. Multiple translations by subpixel distances enable higher resolution images to be realized, with a maximum resolution for the present system of 3k x 3k RGB pixels (which roughly corresponds to high quality 35mm format photographic images). In order to achieve maximum sharpness of the final combined image, the aperture of each pixel is masked down to 1/4 of the pixel area.

Table 1 shows the microscanned operating modes of the MASCOT camera. The basic image mode requires 3 exposures and generates a full 1k x 1k RGB image. By repeating this procedure with subpixel offsets, colour images with 2k x 2k and 3k x 3k pixel resolution can be produced. In each case the higher resolution image is built up from a number of lower resolution but full field basic images.

Table 1 : Microscan operation modes

Mode	Resolution (colour pixels)	Number of scans	Effective pixel pitch (μm)	Equivalent A4 sampling	
				pix/mm	pix/inch
0	340 x 1k	1	-	-	-
1 (basic)	1k x 1k	3	19	3.3	85
2	2k x 2k	9.5	12.7	6.7	169
3	3k x 3k	27	6.3	10	254

Camera head

The colour imager and lens module with microscanning are built into a commercial 35 mm film camera body (Canon F1) which conveniently provides view-finder, exposure control and flash synchronization functions. The CCD imager is connected by a 2 metre flexible cable to the driver electronics and to the video preprocessor which generates a video signal for inputting to the colour processor.

Colour processor

The function of the colour processor is to decode the colour information in the video signal and provide suitable matrixing functions for producing a high quality colour triplet (RGB, XYZ, ...) image. The video signal is first digitized (10-bit, 5 MHz) and all subsequent processing is carried out digitally. For acquiring the basic image, 3 exposures with 1 pixel image translation between them are taken and stored in 3 frame stores. The data is then read-out (demultiplexed) in the sequence required to reconstruct the basic 1k x 1k colour image. The demultiplexed data passes through a programmable input look-up table / matrix / output look-up table, which produces the optimized RGB data and allows simple image processing functions to be implemented. Finally the data for the basic image is transmitted over a high speed video bus to a 8 MB colour image store in the main workstation; using a dedicated processor system, the processing and transmission takes only 150 nsec. Combination of multiple basic images to produce higher resolution is carried out by this workstation.

3. LENS

The lens for the MASCOT camera was specially developed to meet the requirements of high resolution, large field of view and low chromatic aberrations [1]. Figure 3 shows the optical layout of the lens system, composed of 13 optical elements (lens objective), one sheet of glass (BG 38 + dielectric filter) for the microscan device and one cover glass for the CCD sensor.

The main parameters involved in this design are:

- High resolution
- Range of working distances (magnifications)
- Long back focal length
- Low distortion
- High relative aperture
- Low chromatic aberrations

- Optimization to include the blue region (the wavelength range is from 410 nm to 650 nm).

The solution obtained is the best compromise for all these parameters and required the use of special optical glass and considerable computing effort to solve the problems encountered. Table 2 gives the main characteristics of the objective. The lens covers the very wide range of formats, from A7 to A0, by using a mechanical ring to set the magnification together with position adjustment on the reprographic stand to focus. In addition, objects at infinity can be imaged by the same lens. The relative aperture is constant over the whole magnification range.

Focal length	25 mm
Max. F/number	2.8
Min. F/number	16 (manually controlled)
Resolution	A3 and A4 format 60% MTF at 70 cycles/mm
Distortion	A3 and A4 format 2.5%
Vignetting	Max. 20% at F/2.8
Wavelength correction	410 nm to 650 nm
Sensor format	19.456 mm x 19.456 mm
Optimized image format	20 mm x 14.14 mm
Object format	A7 to A0 and object at infinity
Magnification	5.3X to 59X
Working distance	94 mm to 1450 mm

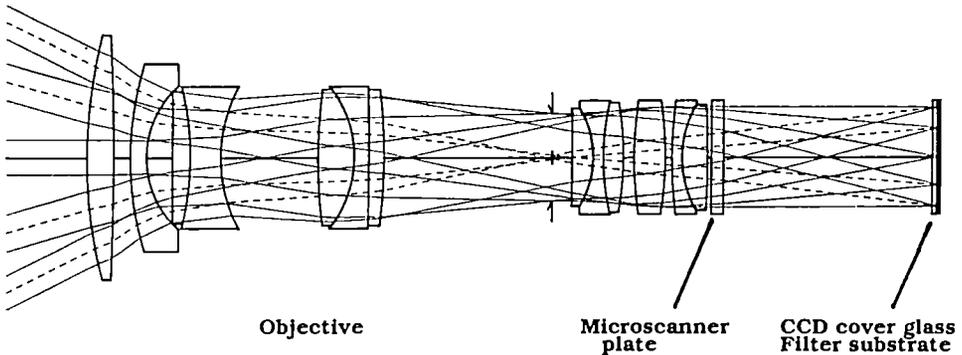


Figure 3. MASCOT lens - optical system

Particular attention was devoted to manufacturing and cost considerations - the opto-mechanical tolerances were carefully studied and optimized in order to minimize fabrication costs. An important feature in this optical system is the possibility to insert the optical microscan component between the last objective element and the CCD sensor. The complete opto-mechanical layout (lens and microscanner) is shown in Figure 4.

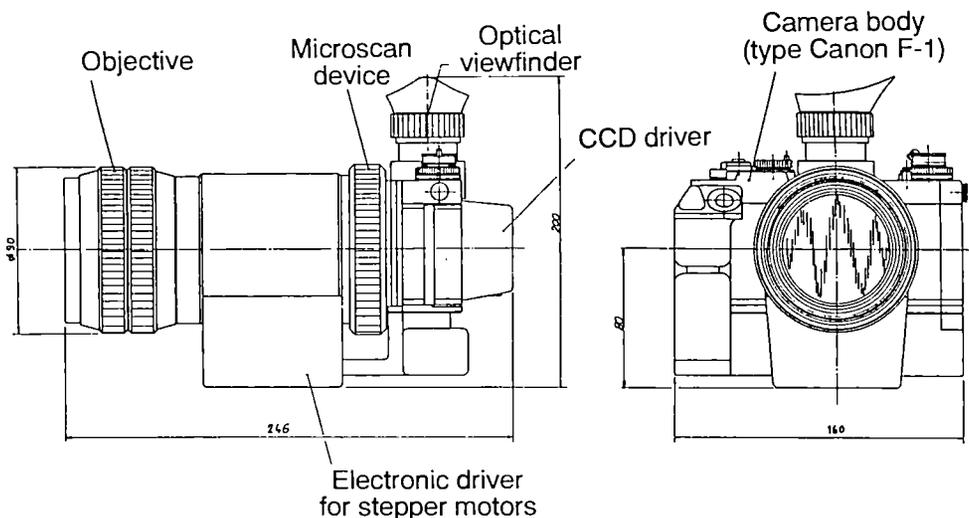


Figure 4. MASCOT lens : opto-mechanical layout

Microscanner

The microscan module can displace the image in the focal plane along 2 orthogonal axes by means of a motorized, tiltable glass plate. The glass refractive index and thickness were chosen to give the required maximum displacement of 2 pixels ($38 \mu\text{m}$). The stepper motors and mechanical arrangement corresponds to about 900 steps for 1 pixel ($19 \mu\text{m}$) translation; the measured reproducibility for the image translation is about $\pm 0.1 \mu\text{m}$.

The required displacement steps and sequence for the acquisition modes shown in Table 1 are commanded by the camera controller. In modes 2 and 3, the image is acquired as a succession of basic mode 1 images (each 3 scans with 1 pixel horizontal translation between exposures resulting in a $1\text{k} \times 1\text{k}$ RGB image), with the required subpixel offsets. A mode 2 image requires four mode 1 images offset in x and y by 1/2 pixel. A mode 3 image requires nine mode 1 images with 1/3 pixel offsets. Mode 2 and 3 images are thus composed of a sequence of images of the complete field but of lower sampling resolution.

4. CCD imager

The CCD imager used in MASCOT project is derived from the THX 31156 imager manufactured by THOMSON TMS []. It is an area array of 1024 lines and 1024 pixels per line (i.e. a 1 Megapixel sensor) in contrast to standard CCD sensors used in camcorders or electronic cameras, which have typically have less than 500,000 pixels. A large number of pixels is one of the major goals of the overall system, in order to get high definition images without moiré effects. Scanners using linear CCDs with a larger number of pixels in a line already exist, but the required scanning in the direction perpendicular to the CCD results in either a long scanning time or very high illumination levels.

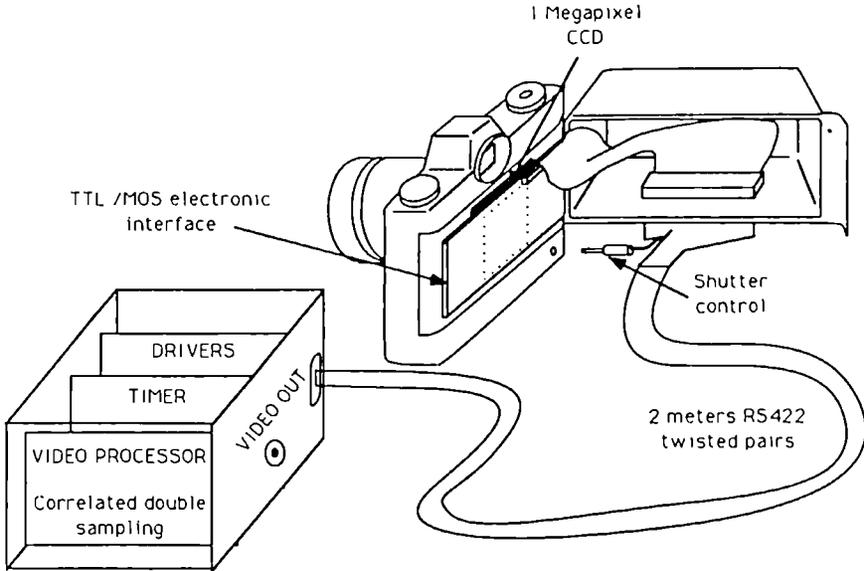


Figure 5. Camera body, CCD imager and driver electronics

The choice of a 1 Megapixel area array is a compromise between cost and performance. To the first order, performance is related to the number of pixels. The price, as for every VLSI device, is related to the area of the chip, which is roughly the product of one pixel area by the number of pixels. The pixel area can be set by the finest available lithography, but also by the sensitivity required by the system: The larger the pixel area the greater the sensitivity. The THX 31156 pixel is a square of $19\ \mu\text{m} \times 19\ \mu\text{m}$. It is realized with a two level polysilicon buried channel technology.

The organization of the array allows a relatively fast read-out time using four outputs, although this feature is not currently used in the MASCOT project in order to simplify the TTL to MOS levels electronic interface. The CCD imager is mounted on a small electronic driver board in a standard CANON F1 camera body (see Figure 5) in place of the usual photographic film. All necessary logic signals and DC supplies are fed to the camera head through a multi-pin connector; the single video output is returned to the main electronics to be processed. This electronics consists of three boards:

- The sequencer, which delivers all the logic signals (derived from a 10 MHz local oscillator) necessary for the CCD.
- The driver board, which lowers the output impedance of the TTL signals in order to supply the camera head with clean signals through two metres of cable.
- The processing board, which performs mainly a correlated double sampling of the video signal coming from the camera head.

At the output of this first processing, the video signals appear as analog samples of 200 nsec. duration whose DC level corresponding to the black level is compared to zero. The signal is then ready to be digitized for further processing.

5. COLOUR FILTER

The MASCOT colour imager consists of the above CCD imager with a superimposed stripe filter to define the aperture and colour response of the pixels. Computer simulation of the camera system was used to optimize the aperture size and shape and the colour filter characteristics. A summary of the stripe filter parameters is given in Table 3.

Table 3 : Stripe filter parameters

No. of pixels	1024 x 1024	
Pixel size	19 μm x 19 μm	
Pixel aperture size	9.5 μm x 9.5 μm	
Aperture shape	square	
Colour filter pattern	Vertical stripes, alternating KLM filters	
Filters	Multilayer dielectric stacks (TiO_2 , SiO_2) designed by simulated annealing ...	
	No. layers	Total thickness (μm)
K ('red')	13	0.74
L ('green')	19	1.35
M ('blue')	19	1.47
Global	12	1.1

Pixel aperture

The programmable resolution offered by the MASCOT camera is achieved by microscanning the scene with the masked pixels of the CCD using selectable distances. Since the size and shape of the apertures in the pixel mask are necessarily fixed and the microscanning is variable, it is essential that the size and shape of the pixel mask apertures are optimized for the anticipated application areas.

Since no complete model of the human visual system is yet available, it was necessary to investigate experimentally the effects of different pixel masks on human observers. To do this, a few representative images were chosen (natural scenes, text of variable resolution, a zone-plate), and a simulation program was written, capable of taking into account also the finite point-spread-function (PSF) of the lens system. The results obtained with these psychophysical experiments can be summarized as follows:

- The geometrical shape of the pixel aperture does not appreciably influence visual quality.
- Optimum visual quality is attained with an aperture size corresponding to the sampling period chosen. If the aperture is about 50% larger than the sampling period, then a simple separable 3-tap post-processing filter exists, capable of improving the quality almost to the perfect situation, albeit at the expense of slightly increased noise.
- Text acquired with 8 bits of gray-value show improved readability compared to binary acquisitions. A4 documents acquired with 170-200 pixels per inch (corresponding to about 2k x 2k pixels for the A4 page) are easily readable and reproducible.

The following conclusions can be drawn from these findings:

- For A4 document acquisition, an important format for the MASCOT camera, the camera will be mostly used in the 2k x 2k mode.
- The optimum pixel aperture, therefore, is a square about $10 \times 10 \mu\text{m}^2$ in size.
- In the 3k x 3k highest-resolution mode, post-processing with the mentioned digital filter is necessary for best visual quality.

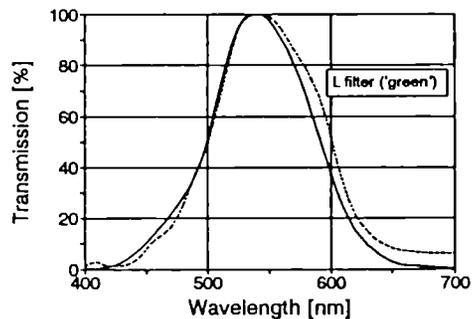
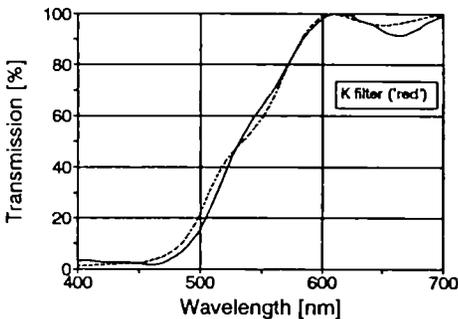
An important feature of the MASCOT camera is the high performance with respect to aliasing in the acquired images. Colour aliasing is absent for all microscanned images, since the microscanned operation is fully equivalent to a 3-chip colour camera. Black-and-white aliasing is dictated by the effective sampling resolution, and is subjectively very small for mode 2 and post-processed mode 3 images.

Colorimetry

The aim of the optimization of the filter transmission curves for the filter was to achieve colorimetric performance beyond that of high performance video cameras and colour scanners (as well as to minimize the thickness of the filters in order to optimize the pattern definition in fabrication). In addition to applications in desktop publishing and technical offices, such a camera would allow fast colorimetric measurements on scenes to acquire colour maps which would be of great value in automated visual inspection of products with coloured textures, e.g. cloth or floor coverings. In all these applications a high level of illumination can be provided. The filter set was therefore optimized for bright light conditions (in contrast to television cameras).

As a key factor for improved colorimetric performance of the MASCOT camera, filter transmission curves and matrixing were optimized together. The optimization of colour filters and matrixing was based on a set of almost 300 test-colours, consisting of the CIE standard test colours, optimum colours at different levels of luminance, and a set of about 200 selected Colorcurve colours [1]. Moreover, since the CCD video output signal is digitized with 10 bit resolution, matrixing is performed very accurately and only rounding errors (mainly due to the 3×8 bit output) are added to the colour signals.

Figure 6 shows the transmission characteristics of the fabricated KLM triplet, which closely match the optimum curves resulting from the simulation. The resulting mean colorimetric error for the test colours is about 1 j.n.d. CIE, which is close to the visual discrimination threshold for colours and significantly better than current commercial (including studio) cameras.



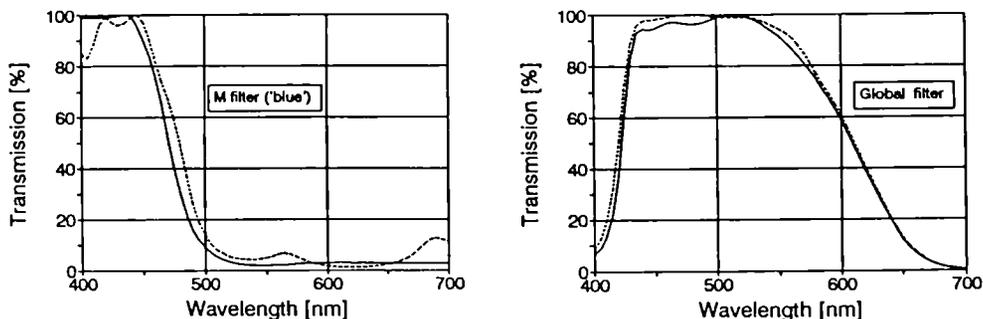


Figure 6. Colour filter transmission characteristics. The broken show the target curves and the solid lines the measured data for the fabricated filters

The major features of the filter design can be summarized as follows:

– **Multilayer dielectric filters**

The use of dielectric filter stacks instead of absorption (dye) filters [1] to fabricate the filters, allowing virtually any desired filter transmission characteristic to be achieved. In practice, the performance of the chosen optimum filter triplet was limited mainly by fabrication considerations (tolerances) for the individual dielectric stacks.

– **Advanced filter design**

The application of advanced filter design techniques using simulated thermal annealing [2] to closely reproduce the optimized transmission characteristics resulting from the simulation work. In addition, the number of layers and the total filter thickness could be minimized, resulting in improved definition and reduced tolerances (improved yield) in the filter fabrication process.

– **Global filter concept**

In order to reduce fabrication tolerances for the patterned stripe filter, an additional band-pass filter covering all pixels of the array was designed to provide the critical short and long wavelength transmission edges. This global filter was deposited on a BG-38 infra-red filter and conveniently incorporated into the microscanner device as the tilt plate.

The result of this work is a set of filters producing very high quality colorimetry and with reasonable fabrication tolerances.

Filter fabrication

The stripe colour filter was realized on a thin glass substrate, subsequently glued onto the surface of the CCD in registration with the CCD pixels (see Figure 7a). This procedure, rather than a direct deposition and processing of the filters on the silicon wafer, allows a more precise definition of the filter patterns and was chosen as a more convenient approach during a prototype phase, allowing the matching the best available CCD and filters.

The KLM filters and the global filter were fabricated by e-beam deposition of TiO₂ and SiO₂ multilayer stacks. The basic structure of the stripe filter is shown in Figure 7b; a summary of the filter structures is given in Table 3.

Global filter

The global filter stack was deposited on a polished 28 mm diameter Schott BG38 (3mm) IR filter substrate. In addition to defining the (critical) low-wavelength and long-wavelength edges of the spectral response, the global filter also functions as the tilt plate in the microscanner module. Since the dielectric filter stack does not require micropatterning, the fabrication of this critical filter is not unnecessarily complicated by subsequent processing steps.

Micropatterned KLM filters

The masked KLM colour filter fabrication involved 4 levels and required the development of new techniques for fabricating large arrays of filter pads with 9.5 x 9.5 mm dimensions. The Cr masks for the lithography were generated from the CCD imager CAD data and fabricated by an external mask shop. The stripe filters were fabricated on 3" disks of Corning 80151 glass (0.5 mm thick), using the following fabrication steps:

Aperture mask level :	Cr film deposited by e-beam evaporation. Aperture definition by resist lithography followed by wet etching.
KLM filter pads :	Pattern definition by polyimide resist. Deposition of filter stack by e-beam evaporation Lift-off to leave required patterned filter level

After completion of the aperture mask and the KLM filter deposition and patterning steps, the stripe filters were annealed at 320°C for 2 hours to produce filter characteristics with good long term stability. The measured transmission curves were in good agreement with the design curves and the stripe filters showed acceptably low defect density (pinholes and other small defects). The completed filters were finally sawn to size for glueing onto working CCD imagers.

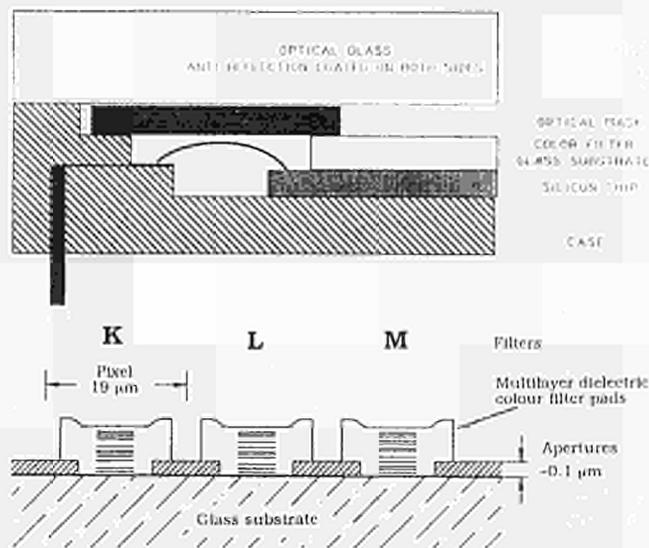


Figure 7. Colour CCD imager - (a) filter mounting on CCD (b) filter structure

Filter mounting

The colour CCD imager structure is shown in Figure 7a. The stripe filter glueing operation is carried out using a UV curing glue. The filter is accurately positioned and aligned to registration marks on a tested, mounted CCD imager using a special tool, pressing the filter onto the CCD surface without bending the glass substrate. The overall flatness obtained for the optical plane - within $15\ \mu\text{m}$ - is low enough to maintain the whole image in the depth of focus of the F/2.8 optics. Great care was taken to minimize flare and stray reflections which might degrade the picture quality. The cover glass is antireflection coated on both sides and an optical shield surrounds the useful optical area to avoid any reflection from the gold-coated pads of the CCD packaging and from the edges of the filter substrate.

6. Colour processor

The function of the colour processor is to separate and transform the signals coming from the sensor to a form suitable for display or printing. In a conventional camera, this is done in the analogue domain - the signal is stored on capacitors and mixed using amplifiers and resistors. Whilst this produces a simple and relatively cheap solution, it is also inaccurate, inflexible and liable to drift (i.e. the system may be correctly aligned one day, but the next day slightly different colours would be produced). The electronic alignment process is difficult and time-consuming to perform, adding great expense to the cost of manufacturing a system. Conventional matrixing has imprecise parameters without real possibility for accurate tuning to new conditions, whereas the MASCOT camera is intended to operate in several environments, each requiring their own set of matrices.

Digital processing

It became apparent early in the MASCOT project that these problems could be eliminated if the processing was performed digitally. This enables the typical problems of the analogue world to be avoided. Once the video signal has been digitized [9], the data can be stored without error, the calculations can be performed accurately and matrix algorithms can be easily modified.

The matrixing operation requires that each of the three output colours RGB is calculated by adding (or subtracting) a fraction of the three input colour signals KLM. This implies three multiplications and two additions per colour of one pixel. Hence an image of $1\text{k} \times 1\text{k}$ RGB would require over 15 million ($5 \times 1\text{k} \times 1\text{k} \times 3$) calculations. This large number of calculations to be performed is the disadvantage of digital processing. Even the most modern personal computer would require several minutes to perform the necessary calculations, which would result in an unacceptably long delay between taking a picture and observing the result. The processing power required could be provided by a super-computer (> 100 million operations per second), but obviously this would be unfeasible. A practical approach is to build a dedicated computer - one which is designed to carry out a single task as opposed to a personal computer which can be programmed to perform many different tasks.

The feasibility of using Digital Signal Processors (DSPs) for MASCOT was studied. Their use would reduce the processing times down from minutes to several seconds. This speed improvement would still not make the system "real time", i.e. it would still take longer to process the colours that it takes to acquire a picture. Also these devices

are still expensive and take a great deal of time designing circuitry and programming before they can be used.

Dedicated hardware

The alternative, which was used for the MASCOT camera, is a fully dedicated machine. Here the flow of the data is chosen to match that of the algorithm. For maximum speed every mathematical operation in the algorithm is computed simultaneously using a dedicated piece of electronics. The additions are relatively easy to perform, however the multiplications are more difficult.

The problem of multiplying numbers can be simplified somewhat as the colour signal needs only to be multiplied by a constant. Of course, this constant will change for different light sources, monitor phosphors, printer dyes, but remains constant for an image. This simplifies the multiplication greatly. As there is now only one variable, a look-up table (LUT) can be used to generate the desired multiplication result. The data is stored in EPROMs (erasable, programmable, read-only memories), which typically can produce a result in approximately 150 nsec. - i.e. 6 million answers per second. Although this may seem fast, the camera's sensor is capable of producing 5 pictures per second requiring a throughput of $5 \times 15 = 75$ million calculations a second in order to produce real-time output. Fortunately the task of processing the colour information can be sub-divided.

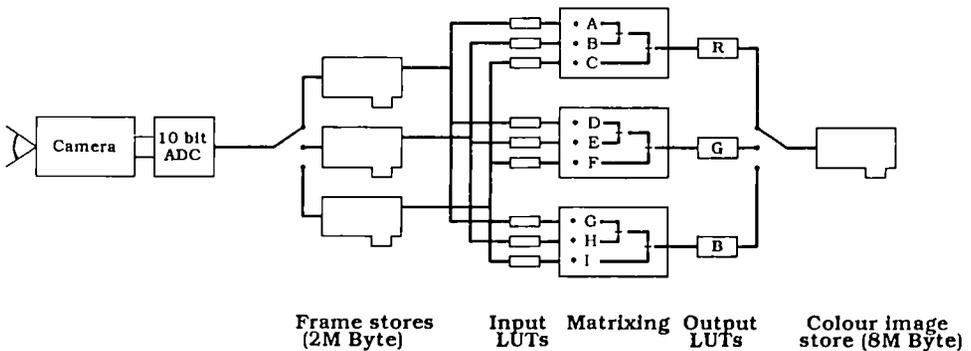


Figure 8. MASCOT camera digital colour processor

A schematic of the MASCOT digital colour is shown in Figure 8. The video signal is first digitized using a 10-bit pixel synchronous ADC [] and the data for the basic image (3 scans) is stored in three 2 MByte frame stores. The processing of each of the three colour channels is identical (apart from the coefficients) and independent. Also, the multiplications required for each channel can also be carried out independently. This feature allows the use of 9 (three per channel, three channels) multiplying units operating simultaneously, increasing the throughput by a factor of 9. The outputs of the multipliers are stored and then summed whilst the multipliers are calculating the next value. This technique is called pipe-lining and using this technique the MASCOT colour processor electronics achieves a throughput of 120 million instructions a second using cheap, readily available components. The use of input and output look-up tables

offers greater flexibility for practical problems such as non-linearity correction of the CCD/ADC stage (input LUT) or gamma correction (output LUT).

7. system performance

The MASCOT camera system as described in this paper has been assembled and evaluated. The subjective assessment of the colour images is that they are overall of very good colour quality and that the resolution improvement obtained using the microscanning is impressive. In particular, the improvement in quality of lines and edges (improved sharpness, removal of colour aliasing effects) between mode 0 images (no microscanning) and mode 1 images (basic microscanned 1000 x 1000 RGB image) is dramatic.

Minor cosmetic imperfections are visible in the displayed images as small (a few pixels in size) spots due to defects on the CCD imager and in the colour filter. Some defects result from the CCD imager fabrication. Most defects probably result from the (filter) fabrication steps, which were not carried out under clean room conditions; they are not inherent and could be avoided by full fabrication and assembly of the colour imager in a clean room. Slight colour shading is visible in dark image areas - this is due primarily to variations in the pixel dark current and sensitivity over the CCD imager and could be reduced by further work on the imager design. Both effects could be largely compensated by suitable LUT operations.

The sensitivity of the MASCOT colour camera can be expressed in terms of the required exposure for an A4 document - for an illumination of 4000 lux with a colour temperature 5400 K (typical for a reprographic stand with fluorescent lamp illumination), an exposure of 1/15 sec. at F/2.8 is required. The colorimetric performance depends upon the kind of illumination (tungsten, line spectra, etc.). Quantitative evaluation of the colour images (100 msec. integration time) gives a measured CIE Colour-Rendering Index of $R_a \sim 88$ (colorimetric error $dE_{CIELUV} \sim 2.7$ j.n.d.) for tungsten (3200 K) illumination with an 80A conversion filter. This is slightly better than that of a good professional broadcast camera. With the experience gained in this work, a further improvement in colorimetric quality is certainly possible - a second round of filter optimization and fabrication should lead to a value close to the predicted 1 j.n.d.

8. Conclusions

The MASCOT colour CCD camera represents a significant advance in colour cameras for desk top publishing and the graphic arts. It is the first camera system described with fully digital colour processing electronics, providing high S/N, stable colour image acquisition, and optimized, multilayer dielectric filters for very high quality colorimetric performance. The unique microscanning approach, with optimized imager pixel aperture masking and an optical microscanning element incorporated in the camera lens, provides programmable resolution up to 3k x 3k colour pixels. The resolution mode can be chosen to match the application requirement. The camera thus forms a flexible, high performance image acquisition system for advanced and professional applications.

9. Acknowledgements

The MASCOT camera development was carried out by a team of workers in an international collaboration project. Valuable contributions of the following are gratefully

acknowledged: C. Appassito, K. Knop, R.E. Kunz, H.W. Lehmann, R. Morf and H. Schütz (Paul Scherrer Institute Zürich, Switzerland); H. Gerber and R. Zumbrunn (Leica AG, Aarau, Switzerland); D. Herault, D. Romand and D. Van Brussel (Thomson TMS, St. Egrève, France); A. Cavanna and S. Viganó (Intel, Milan, Italy); G. Coli and G. Fogaroli (Olivetti, Ivrea, Italy), J. Gordon (SSS, Bari, Italy).

Funding for this project was provided in part by the EC (ESPRIT II Project No. 2103) and by the Swiss Kommission zur Förderung der wissenschaftlichen Forschung (KWF-Projekt Nr. 1804-2).

10. References

- [1] LENZ, R. (1989). Digitale Kamera mit CCD-Flächensensor und programmierbarer Auflösung bis zu 2994x2320 Bildpunkten pro Farbkanal. Informatik-Fachberichte, Vol. 219, Proc. 11. DAGM-Symposium 1989, Hamburg, Oct. 2-4, Springer Berlin ISBN 3-540-517748-0, pp 411-415.
- [2] Faggiano, A., Gadda, C. and Moro, P. (1986). Contemporary Optical Instrument Design, Fabrication and Testing. Proc. SPIE, Vol. 656.
- [3] FAGGIANO, A. (1990). Automatic Lens Design with Pseudo-Second Derivative Matrix : A Contribution. Appl. Opt., Vol 19, p. 4226.
- [4] Product of Thomson Composants Militaires et Spatiaux, St. Egrève, France.
- [5] Colorcurve Master Atlas, Colorcurve Systems Inc. 1988.
- [6] Aschwanden, F., Gale, M.T., Kieffer, P. and Knop, K. (1985). Single-Chip Color Camera using a Frame Transfer CCD. IEEE Trans. El. Dev., Vol. ED-32, p. 1396.
- [7] Morf, R. and Kunz, R.E. (1988). Dielectric filter design by simulated thermal annealing. Proc. SPIE, Vol. 1019, p. 211.
- [8] Morf, R. and Kunz, R.E. (1990). Dielectric filter optimization by simulated thermal annealing: a simulated zone-melting approach. Proc. SPIE, Vol. 1270, p. 11.
- [9] Raynor, J.M. and Seitz, P. (1990). The Technology and Practical Problems of Pixel-Synchronous CCD Data Acquisition for Optical Metrology Applications. Proc. SPIE, Vol. 1395, p. 96.

FAULT TOLERANT DATA MANAGEMENT IN PORDOS (*)

Thomas Becker
Computer Science Department, University of Kaiserslautern
P.O. Box 3049
D-6750 Kaiserslautern, Germany
email: tbecker@informatik.uni-kl.de

(*) Part of this work was funded by a cooperation project with TELENORMA Bosch Telecom.

SUMMARY

In a distributed system, data segments can be managed in a fault tolerant way if they are replicated in several identical copies and distributed on data management servers running on different processors. If a server or a processor fails, copies of the data segments may be lost. To preserve fault tolerance during the lifetime of the system, for each data segment copy which is lost due to a server failure, a new copy has to be regenerated on one of the remaining servers. Therefore, all existing servers must be kept under surveillance to detect server or processor crashes as soon as possible. In this paper we describe two solutions for this surveillance problem. Both approaches are based on an election algorithm which has to cope with process and communication failures. The election algorithm is presented in detail. The protocols proposed in this paper have been developed as part of a distributed operating system which serves as a base for a distributed telecommunication control system.

1. Introduction

In this paper we discuss a solution for a fault tolerant data management problem. In a computing environment which lacks access to stable storage, data segments have to be made available on program demand with a high degree of reliability. Node crashes must not cause any of the data segments to be lost.

In the remainder of this section, a project overview is given, and the distributed operating system kernel PORDOS which is used in the project is described. In the second section the actual fault tolerance problem is presented in more detail. Section 3 discusses two approaches for server crash detection, and in Section 4 we present a fault tolerant election algorithm which is the major part of the server crash detection scheme. Finally, we summarize our results in Section 5.

1.1. Project Overview

Within the ESPRIT II program the three year project DAMS ("Dynamically Adaptable Multiservice System") has been in progress since April 1989. The project consortium consists of TELENORMA Bosch Telecom (Germany), BNR-Europe (Great Britain), JS TELECOM (France), and several associated partners and sub-contractors.

The DAMS project was conceived to meet the requirements for future private communication and office systems (12, 16, 17). DAMS deals with the aspects of service integration and the demand for more sophisticated terminals, offering a flexible access capability for services with variable bandwidth exceeding 64 kbit/s. The major work of

the project is being directed towards the construction of a prototype which is to be demonstrated on a suitable test site. The most important goals of the DAMS project can be summarized as follows:

- Integration of services
In future systems for the business communication market the integration of existing services such as delay-sensitive telephony services and time-independent data services, is an important requirement. Customers will be provided with a single communication infrastructure as opposed to the completely distinct worlds of voice and data communication coexisting today.
- Flexible use of bandwidth
The variety of supported services requires a dynamic allocation of bandwidth to provide an effective means of efficient usage of bandwidth, and hence to increase system performance. Variable bit rates needed by new services such as bandwidth on demand terminals have to be taken into account.
- Adaptability to changing user demands
The introduction of new services should be feasible after the installation of the system. It must be possible to add or remove services and to reconfigure a system according to a new customer profile.
- Increased reliability and availability
System and network management must ensure continuing operation of the system in the presence of specific faults and failures, in accordance with accepted principles of graceful degradation. Since the provision of fault tolerance causes overhead in terms of execution time, memory usage, or additional redundant hardware devices, it is desirable that the degree of fault tolerance can be selected by customers.

The DAMS system consists of autonomous subsystems which can operate in a stand-alone mode or can be connected by a backbone system to build larger networks. The subsystems are subdivided into the following main components (Figure 1): a local switch unit (LSU), a backbone unit (BBU), a subsystem control unit (SSCU) and several port units (PU). The LSU establishes and maintains the connections between various system terminations, while the port units serve as interfaces to the external environment. External devices can be telephones, bandwidth on demand terminals, local area networks (LAN), main frame hosts, etc. The Backbone is based on the FDDI-II standard and uses a token ring protocol in conjunction with an optical fibre transmission medium. The SSCU is responsible for executing the overall control software, namely the subsystem management, major parts of call control and various services of a distributed operating system. The unit dependant parts of the control software are executed by the respective units.

From a conceptual point of view all port units and the subsystem control unit run the distributed operating system kernel PORDOS, whose basic concept are described in the next section. (For the prototype system to be developed within the project this is only true for the ISDN port unit and the subsystem control unit.)

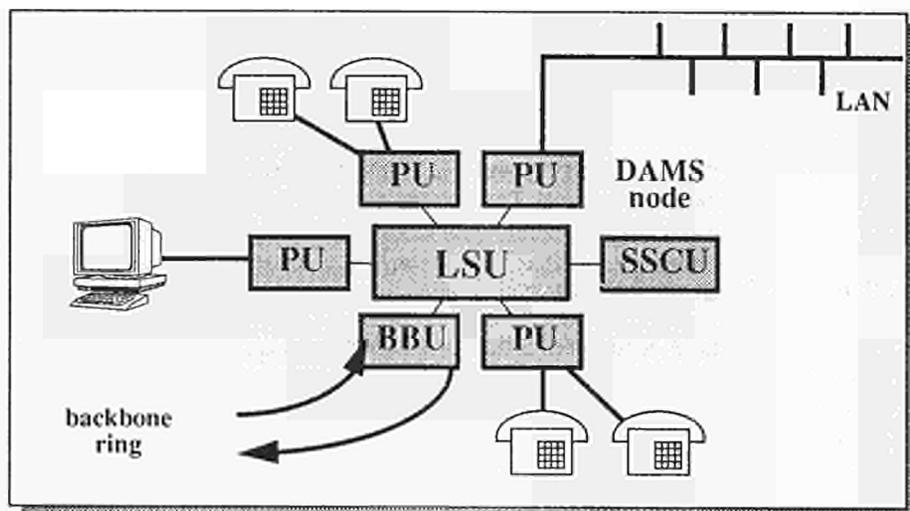


Figure 1: Structure of a DAMS node

1.2. The Distributed Operating System Kernel PORDOS

The distributed operating system kernel PORDOS (*portable real time distributed operating system*) (15) forms the base for high level services of the distributed operating system used in the DAMS nodes. The PORDOS kernel has been designed with respect to the requirements of a modern distributed operating system kernel as well as to its ability to cope with soft real-time conditions.

Applications running on top of the PORDOS kernel are structured as teams. A team consists of a set of processes which share a common address space. Processes can be created and destroyed dynamically in a team. Using the common address space of a team, inter-process communication within the team can be very fast and efficient. For the communication between different teams PORDOS provides a more general and powerful message transmission service. Messages are of fixed length in order to avoid the need of dynamic memory allocation for a message buffer at the receiver's site. A message is sent to a port (not to be confused with the port unit of a DAMS node) which can be created by any process of a team. The port is owned by the team, so that every process of this team can actually receive a message at the port, independent of which process has created the port. Sending a message to a port is only possible if the network-wide unique port identifier is known to the sender. A name service can be used to obtain port identifiers of services which are only known by name.

By choosing one of three different protocol types, the application programmer can determine the message semantics and the synchronization of its delivery. A datagram message can be sent to its destination without any synchronization of sender and receiver. The sender will not be informed if the message has arrived at its destination or not. In a more sophisticated message transmission protocol the sender is guaranteed that the message has arrived at the destination port. This protocol still does not enforce any synchronization between sender and receiver, since the message is not yet received by a process. The third protocol performs a remote procedure call (RPC), where sender and receiver are forced to synchronize each other. The sender will be blocked in the send operation until the receiver has sent a reply message. An

"at-most-once semantics" (5) is guaranteed by the protocol to prevent a service request from being processed multiple times due to message losses.

Each of these protocols can be used for sending a message either to a single port or to a group of ports. By the latter, PORDOS provides a multicast facility which is an important mechanism for the design of fault tolerant applications. In case a request message is sent to a port group using an RPC, the sender will be deblocked as soon as the first reply message has arrived. The sender can wait for further reply messages if needed.

2. The PORDOS Program Service

Depending on the configuration of a port unit (which may change during run time), different application teams have to be executed on the port unit's processing element to provide different functions. In case the configuration has to be changed, e.g. by adding a new device function, a new application team responsible for this function has to be created on the port unit. The PORDOS Program Service is used to provide the new team with the code it is supposed to execute.

2.1. Creating a New Team in PORDOS

To create a new team in PORDOS, two steps have to be performed. First, the parent team (i.e. the team which will create a new team) sends a request to the PORDOS Program Service containing the (unique) name of the code segment which has to be executed by the new team. This request is sent to the Program Service as an RPC message. The reply message of the request will contain a port identifier from which the requested code segment can be downloaded.

In the second step, the new team is actually created by the parent team. An initial process, the boot process, is started automatically in the new team. The boot process sends an RPC message to the Program Service port which was passed to the new team as a parameter. The reply message from the Program Service contains the total size of the code to be downloaded. After having allocated enough memory, the boot process can request the code segment from the load port of the Program Service, copy it into its own address space and start the new team.

2.2. Requirements for the Program Service

The concepts of the PORDOS Program Service are determined by two major requirements. First, the Program Service does not necessarily have access to stable storage. In the overall system concept, file servers are available only on few DAMS nodes, while most of the subsystems do not have any non-volatile storage devices connected to them. To prevent that code segments have to be downloaded from these file servers each time a new team is created, and to increase the availability of the code segments, the segments are kept in main memory. Due to the limited storage capacity, it is impossible to store all existing code segments on a single processor. As a consequence, the segments have to be distributed on several processors.

To achieve the distribution of the code segments, a number of Program Servers is running on several processors of the DAMS system. All servers together provide the Program Service for the user, who can request this service by sending messages to the Program Service port group. The Program Service then determines which server

will be responsible for processing the user request. Choosing this server is completely transparent for the user.

A second requirement for the Program Service is its reliability. Code segments are implementations of device specific functions which are available at a port unit. Since a subsystem must be reconfigurable at any time, it has to be guaranteed that these functions are available somewhere in the system, even after a site managing the code segments of the functions has failed. However, since code segments are managed in (volatile) main memory, they will be lost after a node crash. It is the responsibility of the Program Service to ensure that the overall system functionality (i.e. the number of available code segments) is not reduced by failing sites.

2.3. Fault Tolerance Principles of the Service

In order to tolerate server failures (e.g. due to processor crashes), the PORDOS Program Service manages each code segment in several identical copies on different Program Servers (2,14). The more copies of a code segment are available in the system at a time, the more site failures can be tolerated. In our case, the servers agree upon a system wide constant N which determines the *minimum* number of existing copies of each code segment. Whenever a server fails, this lower bound may be violated for any of the segments managed by the failed server. To avoid the complete loss of all segment copies due to subsequent server failures, copies of lost segments must be restored on the remaining servers until all segments are available in N identical copies. Therefore, server crashes have to be detected as soon as possible.

In addition to server crashes, the protocols described in this paper are able to cope with lost messages. The protocols assume that messages arrive at their destination within a fixed time. If a message fails to arrive within this time, it is assumed to be lost. Messages are received at a port in the same order as they were sent.

3. Server Crash Detection

Each Program Server consists of a (dynamically changing) collection of service processes and one additional surveillance process. A new service process is created whenever a code segment is requested from a parent team which wants to create a new team. The process will be terminated as soon as the new team has downloaded the code segment.

The surveillance process is invisible to the service processes. Its only task is to communicate with the surveillance processes of the other Program Servers to detect server crashes. In the remainder of this paper, we therefore neglect the existence of further processes and restrict our considerations to the surveillance process only. For brevity, the surveillance processes are simply called "processes". It is assumed that in case of a crash the whole team fails. Failures of a single process within a team are not considered.

To detect server crashes, we describe two different approaches. In the first approach, a central *crash detection manager* periodically multicasts polling messages. The second approach is based on the idea of a *virtual token ring*.

3.1. Central Crash Detection

Before the surveillance of the Program Servers can be started, one of the surveillance processes must be elected to be crash detection manager. This crash detection manager is responsible for multicasting polling messages periodically to all other

processes under surveillance. The other processes are expected to reply immediately to these polling requests with an "I am alive" message. If a process response is not received after three tries, the crash detection manager assumes that this process has crashed. Since the crash detection manager may fail as well, the other processes have to control the periodical arrival of the polling messages. Thus the failure of the crash detection manager can be detected if the polling messages cease to arrive at the other processes for several subsequent periods. If the crash detection manager has failed, the remaining processes have to start another election to determine a new crash detection manager.

Electing the crash detection manager is the major problem of central crash detection. Though the election problem is fundamental in distributed systems and many different election algorithms based on a variety of computational models have been developed in the past, the majority of these algorithms neglect the possibility of communication and process failures. In Section 4 we will present an election algorithm with which a crash detection manager can be determined despite communication and process failures.

3.2. Distributed Crash Detection

In our distributed approach of crash detection, all processes are organized in a virtual ring so that each process knows the address of exactly one neighbor. A token is passed from one process to the other. Whenever a process crashes, the token will eventually be lost. This loss can be observed by the remaining processes which expect the token to arrive within a predefined time (the token circulation time).

If the token is lost, a new one has to be generated. Further, if the loss was caused by a process crash, the virtual ring is broken and must be re-established. For this purpose, a *ring manager* is selected. This ring manager includes new processes into the ring on demand and is responsible for generating the new token. The ring manager is selected by using the election algorithm described in the next section.

3.3. Other approaches

More sophisticated membership protocols proposed by Cristian (8) provide each processor with a consistent view of all other operating processors. However, these protocols make use of atomic broadcast mechanisms (3,5,7) which guarantee broadcast messages to arrive at all receivers in the same order, at the cost of additional control messages. With these protocols it is possible to determine the exact number of servers currently operating in the system.

For our protocols we do not need atomicity when multicasting messages. However, these protocols are not able to determine the exact number of servers, but rather a *minimum* number, which is sufficient for the requirements of the Program Service (and, in fact, for many other applications).

4. A Fault Resilient Election Algorithm

We determine a crash detection manager or a ring manager by an election algorithm which is able to cope with communication and process failures.

Fischer et al. (9) have proven that it is impossible to reach consensus in a distributed system (and hence agree upon a *single* winner) if at least one process fails during the

execution of an election algorithm. In (13), Fischer's results have been extended to a large family of distributed algorithms.

Due to these results, a number of election algorithms have been developed which are applicable for more realistic computational models. Abu-Amara (1) describes an algorithm for a computational model in which some communication links between processor pairs may be faulty (i.e. messages are lost), while the processes themselves are not allowed to fail. This algorithm has been modified by Itai et al. (11) so that process failures are tolerable as long as they occur before the election algorithm is executed. The same assumption is made by Bar-Yehuda et al. (4) who describe a fault-resilient method for constructing a spanning tree, which is an election-related problem.

All these algorithms cannot be applied in a distributed system where both communication subsystem and processes may fail at an arbitrary time. Due to the impossibility result (9) a different approach for electing a winner in a set of processes has to be found.

In our approach, the notion of the term election is relaxed in a way that we demand our election algorithm to find *at least one* winner, provided that one of the processes survives. The election can be initiated by an arbitrary number of processes at a time. Of course, trivial solutions (e.g. declaring every process a winner) are not considered. The goal of our algorithm is to find exactly one winner if no errors occur, and as few winners as possible in the presence of errors.

In both surveillance concepts described above the co-existence of several winners can be tolerated. In case of the central crash detection, the polling messages are multicast by more than one crash detection master in parallel. If several ring managers are elected in the distributed approach, each of them may construct its own virtual token ring. However, though in both cases some additional communication overhead may arise, no process will be excluded from the surveillance.

Garcia-Molina has described an Invitation Election Algorithm (10) which also finds at least one winner. In this algorithm one or more processes form a group with one distinct leader. Initially, each process forms its own group, with itself as the only member and leader. A leader periodically invites other groups to join his group, thus increasing the group size. Obviously, this algorithm starts with n leaders (where n is the total number of processes) and successively reduces the number of leaders and groups by merging groups until a single group with a single leader remains. However, communication failures (e.g. due to network separation) can result in the existence of several leaders.

4.1. Description of the Election Algorithm

For performance reasons, the datagram message transmission protocol in conjunction with the multicast facility of the PORTOS kernel is used in the election algorithm. In some cases, the sender of a multicast message expects a reply message to be sent back by a potential receiver. Since the sender does not know whether the reply message will eventually arrive (e.g. because the sender is the only process in the system), it must bound the waiting time for the reply. A reasonable minimum time bound is given by twice the assumed maximum transmission time of a message, plus the time used by the receiver to process the message.

Immediately after multicasting a message, the sender starts a *reply timer (RT)* which runs for a fixed time T_{RT} . If a reply message arrives before the timer runs out, it is accepted by the sender process. If it arrives after the timer has expired the reply message is ignored and assumed to be lost.

The algorithm uses two different types of multicast messages: a `WHO_IS_MASTER` message (often called query message) and an `I_AM_MASTER` message (used as reply message). Both message types have fixed length and contain the unique identification of the sender process.

4.2. Basic Principles

To present the idea on which our algorithm is based, we first assume that the communication subsystem guarantees that no messages are lost. This restriction will be removed in the next section.

A process initiates the election by multicasting a `WHO_IS_MASTER` message. If no `I_AM_MASTER` message arrives before the reply timer expires, the process declares itself winner and multicasts an `I_AM_MASTER` message.

If, however, a reply message from the current winner is received, the process notices that it has lost the election. In the remainder of this paper, the time between sending the `WHO_IS_MASTER` message and the reception of an `I_AM_MASTER` message (or sending the winner message in case of an election triumph) is referred to as the *election phase* of the process.

Consider a process which has already finished its election phase and now receives a `WHO_IS_MASTER` message. If it has won the election in its own election phase some time ago, it sends back an `I_AM_MASTER` message and thereby terminates the election phase of the process which has sent the query request. Otherwise the query message is ignored.

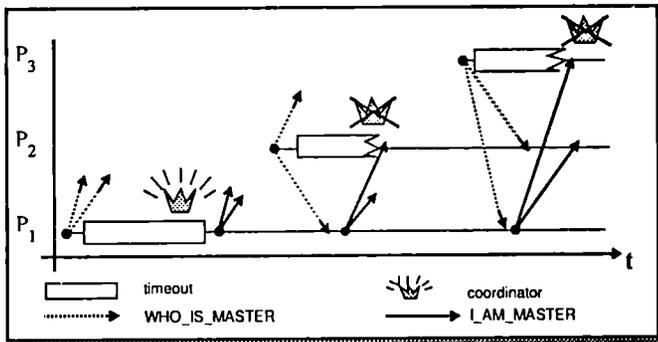


Figure 2: Space-Time Diagram of the Election Phase

Figure 2 illustrates this simple scheme by a time-space diagram. First, assume that a winner (process P1) has already been elected, and now processes P2 and P3 start up. Both multicast their `WHO_IS_MASTER` messages (indicated by dotted arrows) and eventually receive an `I_AM_MASTER` message as a reply (indicated by a solid arrow) from the winner P1. Notice that the reply messages from the winner are multicast to all other processes. A reason for this is discussed below.

The mechanism described so far works as long as the processes start the election algorithm one at a time, which means that the election phase of one process is finished before another process starts its election phase. We now examine the case where several processes initiate the algorithm so that their election phases overlap, as shown in Figure 2. In this case, the request messages of these processes will cross each other on their way through the network, which means that a process may receive other query

requests while waiting for a reply message. If a winner already exists, it will answer the query requests by multicasting an `I_AM_MASTER` reply message, thus terminating the election phases of the senders.

Suppose that no winner has been elected yet. Then no reply messages will arrive at the senders, causing their reply timers to expire. Each of the processes may now conclude that it has won the election. To avoid this, a process waiting for a reply message has to consider `WHO_IS_MASTER` messages from other processes which arrive during its election phase:

- if a query message from a process with a higher process identification (which is included in every message) arrives, the receiving process stops its reply timer and waits for an `I_AM_MASTER` message. The mechanism which avoids an infinite waiting time will be described later.
- query messages from processes with a lower identification are ignored.

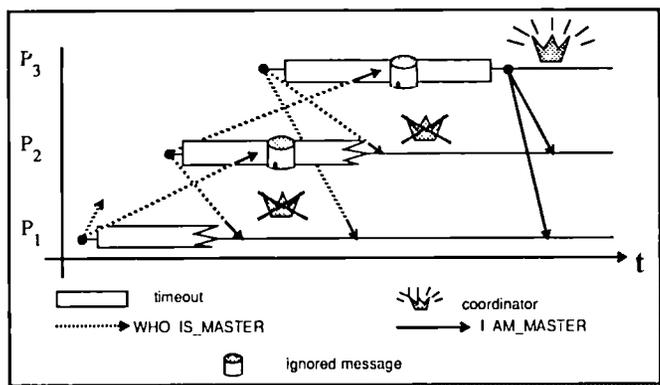


Figure 3: Multiple Processes Starting Simultaneously

Figure 3 depicts an example in which several processes start the election algorithm with overlapping election phases. Let " $>$ " denote a total ordering of the processes based on their (network-wide unique) process identifiers, and $P_3 > P_2 > P_1$. The query requests of P_1 and P_2 propagate through the network. When P_1 receives the request from P_2 , P_1 notices that a process with a higher identification also competes for the winner and therefore gives up (indicated by the broken timeout rectangle).

When receiving the `WHO_IS_MASTER` message from P_1 , P_2 compares the sender's identification with its own. Since the sender's identification is lower, P_2 ignores the message. Subsequently a request message from P_3 arrives at P_2 (and at P_1), which forces P_2 to give up its competition. The request from P_2 is ignored by P_3 , because of $P_3 > P_2$. Finally, the reply timer of P_3 expires, and P_3 can now multicast its election triumph "`I_AM_MASTER`".

In this example it becomes obvious why the `I_AM_MASTER` message of P_3 must be multicast. Process P_3 knows that P_2 has given up, but it does not know that P_1 is also waiting for an `I_AM_MASTER` reply, since P_3 did not yet exist when P_1 had multicast its `WHO_IS_MASTER` message. Therefore, to ensure that all processes waiting for a reply message eventually will receive it, the `I_AM_MASTER` message must also be multicast.

4.3. Unreliable Communication and Process Failure

In the discussion above we have neglected the possibility of message losses and process failures. The problems arising from an unreliable communication subsystem and process crashes are the subject of this subsection.

Consider a process which starts the election after a winner has been chosen. Assume that this process does not receive an answer from the winner before its reply timer expires, either because its `WHO_IS_MASTER` message did not arrive at the winner, or the winner's `I_AM_MASTER` message was lost. The process will therefore declare itself winner of the election. This leads to a situation where more than one winner exist at the same time.

We now discuss the cases where processes involved in the election fail. It is obvious that the failure of a process which cannot become winner of the election is not critical. However, if the process which is going to win the election fails immediately before multicasting the `I_AM_MASTER` message, other processes which are waiting for the arrival of this message may be blocked forever.

To prevent a process from being blocked infinitely, another timer must be started whenever a `WHO_IS_MASTER` message arrives. This timer is called *crash detection timer (CDT)* and runs longer than the reply timer (i.e. $T_{CDT} > T_{RT}$). If the crash detection timer of a process expires, the process stops waiting for an `I_AM_MASTER` message and starts the election anew. The arrival of an `I_AM_MASTER` message stops the timer and terminates the election phase of the process.

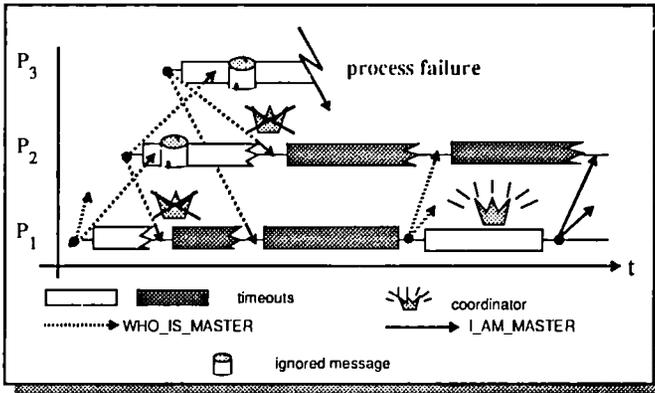


Figure 4: Process Crash During the Election Phase

In Figure 4 the example of Figure 3 is used to illustrate this mechanism. The crash detection timer is represented by dark rectangles. Process P₃ is assumed to fail before it can multicast its election triumph. Since no `I_AM_MASTER` message is multicast after P₃ has failed, the CDT of process P₁ eventually expires, causing P₁ to multicast another `WHO_IS_MASTER` message and to start its reply timer. On the arrival of this message, P₂ restarts its CDT. Finally the reply timer of P₁ runs off and P₁ wins the election.

4.4. Formal Description of the Algorithm

In the following, the election algorithm is presented in a C like notation. It is assumed that a timer service is available which runs in the background and the timers provided by this service expire asynchronously. The function `set_timer (i, t)` sets up a timer

named i which runs for t time units. It does *not* block the calling process until the timer has expired. If timer i expires, the statement sequence following the `on_timeout (i)` statement is executed. Otherwise this statement sequence is ignored. An `unset_timer (i)` function call clears timer i .

function `election ()` returns `ProcessId`

```

{
    winner = None;
again: multicast ( WHO_IS_MASTER, my_proc_id );
    set_timer ( RT, TRT );
    on_timeout ( RT ) {
        winner = my_proc_id;
        multicast ( I_AM_MASTER, my_proc_id );
        return ( winner );
    }

while ( True ) {
    receive ( msg );
    if ( msg.type == I_AM_MASTER ) {
        /* winner exists */
        winner = msg.sender_id;
        return ( winner );
    }
    else {
        /* msg.type == WHO_IS_MASTER */
        if ( msg.sender_id < my_proc_id ) {
            /* ignore message */
        }
        else {
            /* msg.sender_id my_proc_id */
            /* message from process with higher id has
            arrived */
            unset_timer ( RT );
            while ( True ) {
                /* set up crash detection timer */
                set_timer ( CDT, T_CDT );
                on_timeout ( CDT ) {
                    goto again;
                }
                receive ( msg );
                if ( msg.type == I_AM_MASTER ) {
                    /* winner exists */
                    winner = msg.sender_id;
                    unset_timer ( CDT );
                    unset_timer ( RT );
                    return ( winner );
                }
                else { /* msg.type == WHO_IS_MASTER */
                    /* unset crash detection timer and go
                    to beginning of loop */
                    unset_timer ( CDT );
                } /* else */
            } /* while */
        } /* else */
    } /* while */
} /* end function election */

```

4.5. Correctness and Complexity

In this section upper bounds for the message and time complexity of the algorithm are determined with the help of two lemmas. Their proofs are sketched in an informal way. In the first lemma, we examine the algorithm under the assumption that no errors occur. The second lemma considers the correctness of the algorithm in the presence of both communication and process failures.

We assume that appropriate timer values T_{RT} and T_{CDT} are chosen to guarantee that a message can propagate through the network, be processed, and return to its sender, and that the condition $T_{CDT} > T_{RT}$ is satisfied.

Lemma 1: If no errors occur during the election algorithm, a single winner is found in $O(n)$ time units using $O(n)$ multicast messages, where n is the number of processes involved in the election.

Proof: We first prove that the algorithm terminates. Assume the contrary. The only reasons for the endless execution of the algorithm can be:

- an infinite loop of computation. This can be explicitly excluded by examining the program text since there are no loops which do not contain a receive statement.
- a message receiving operation which blocks forever because no messages arrive. Before each receive operation of the algorithm, a timer is started. The operation will be aborted if the timer expires. Therefore, a process cannot be blocked infinitely in a receive operation.
- an infinite, alternating sequence of computation and receive operations. Such a sequence can occur only if the timer RT has been aborted by a `WHO_IS_MASTER` message from a process with a higher identification, and CDT is restarted again and again. Notice that since $T_{RT} < T_{CDT}$ and the assumption that neither messages are lost nor processes fail, either a `WHO_IS_MASTER` message from a process with a higher identification, or the `I_AM_MASTER` message from the winner must arrive before the timer CDT runs off. Hence the sequence of periods in which CDT is restarted and aborted could be infinite only if an infinite number of processes started the election by sending a `WHO_IS_MASTER` message, which again contradicts our assumptions.

Hence no possibility exists for endless execution, and the algorithm terminates. We now prove that after termination a single winner is selected, and that every process in the network eventually knows the identification of the winner.

Assume that more than one winner is selected by the algorithm. In each winner process the timer RT must have run out before a `WHO_IS_MASTER` message from a process with a higher identification or an `I_AM_MASTER` has arrived. Without loss of generality we restrict our proof to only two winners, P_1 and P_2 . The proof for more than two winners is straightforward.

Consider the case where the timers RT of both processes P_1 and P_2 run out at nearly the same time. Since P_1 and P_2 have multicast their `WHO_IS_MASTER` messages before starting the timer and no messages are lost, the processes must have received each other's `WHO_IS_MASTER` messages. However, since the identifications of P_1 and P_2 must be different, one of them (the process with the smaller identification) must have stopped its timer RT to wait for an `I_AM_MASTER` message, which contradicts our assumption above.

Now assume that a process P_1 has declared itself winner before the other process P_2 starts its election algorithm. Since no messages are lost, the `WHO_IS_MASTER` message of P_2 arrives at P_1 . P_1 in turn responds with an `I_AM_MASTER` message, which must arrive at P_2 before its timer runs off. Therefore, P_2 cannot become winner, and P_1 must be the only winner in the network.

It was shown that the election algorithm determines a single process as a winner in finite time. We now discuss the message and time complexity.

For the complexity analysis it is assumed that a multicast message is as expensive as a point-to-point message, independent of the number of receivers or the physical distance between sending and receiving site. A message is supposed to be delivered after t time units.

If the first process comes up and starts the election exclusively, two multicast messages are required: one for the `WHO_IS_MASTER` message, and one to multicast its election triumph. Each process which joins the system separately sends out one `WHO_IS_MASTER` message and finally receives the `I_AM_MASTER` message from the winner. Thus, if n processes start up one by one, $2n$ multicast messages are needed to determine the winner of the election.

If n processes start up with overlapping time intervals (i.e. each process starts before the timer RT of any other process has run off), each of the n processes will send out a `WHO_IS_MASTER` message to query for the winner. Those processes which receive a `WHO_IS_MASTER` message from a process with a higher identification will abort their timers and wait for an `I_AM_MASTER` message without sending any further messages. In fact, all processes but one (which finally is going to win the election) will receive at least one `WHO_IS_MASTER` message from a process with a higher identification, so that only one `I_AM_MASTER` message will be sent. This results in a total amount of $n + 1$ multicast messages.

The time complexity of the election algorithm is influenced by the duration of the timer RT (the duration of CDT is relevant only in case of failures). The time for processing messages is neglected in the analysis. Since the election algorithm does not force all processes to take part in a single election phase, it is impossible to determine how long it takes to elect a coordinator out of n processes if they start with arbitrary time offsets. Therefore it is assumed that initially no winner exists and n processes initiate the election with overlapping election phases.

Obviously, the least time is required by the algorithm if all processes start up at the same moment. All processes except the one with the highest identification receive `WHO_IS_MASTER` messages from stronger processes, and therefore cease the election competition. After the time-out of the process with the highest identification has expired, this process sends the `I_AM_MASTER` message. The total time consumed in this case is $2t + T_{RT}$, independent of the number of processes involved in the election.

In the worst case, the processes start up in increasing order (according to their identification numbers). Let P_1, \dots, P_n denote the starting processes and $P_{i+1} > P_i$ for each $i \in \{1, \dots, n-1\}$. The election algorithm will require the maximum time if for each i the request message of P_{i+1} arrives at P_i immediately before the timer of P_i expires, causing P_i to give up the competition. The processes have to start with equal time offsets of $T_{RT} - \tau$. The last process and final winner, P_n , starts $(n-1)(T_{RT} - \tau)$ time units after P_1 . Eventually, the timer of P_n expires, and an `I_AM_MASTER` message is multicast. The total amount of time in this worst case is

$$(n-1)(T_{RT} - \tau) + T_{RT} + \tau = n(T_{RT} - \tau) + 2\tau.$$

Lemma 2: In the presence of errors, the election algorithm terminates and determines at least one winner if one of the processes involved in the algorithm survives.

Proof: Assume that the algorithm does not terminate. As shown in the proof of lemma 1, the timer CDT of a process must be restarted infinitely often. There can be two reasons for the timer restart:

- CDT can be aborted by the arrival of an `I_AM_MASTER` message. In this case the algorithm terminates.
- a `WHO_IS_MASTER` message has arrived. Any `WHO_IS_MASTER` message (with the exception of the ones that are sent by each process initially) is sent after the CDT timer of another process has expired. This means that either the `I_AM_MASTER` message of the winner was lost, or the winner has crashed before it could send the message. In both cases, a new election cycle is started by the process whose timer ran off. Assume that all subsequent `I_AM_MASTER` messages of the winner - if it still exists - will not arrive at the process to be considered (if they do then the algorithm terminates). The winner will not send a `WHO_IS_MASTER` message either. Therefore, in the new election cycle only $n-1$ processes take part. Each time an error occurs, the number of processes performing the election algorithm decreases. In the trivial case where $n=1$ the process terminates since no `WHO_IS_MASTER` message can force the timer CDT to be started. Hence only a finite number of `WHO_IS_MASTER` messages can cause the CDT timer to be restarted.

We now show that the algorithm determines at least one winner and that each process knows the identification of the winner. Assume that no winner is selected and every process has finished the algorithm. Then either timer RT or timer CDT of each process must have been aborted by receiving a message (if RT runs off the process declares itself winner; if CDT expires a new `WHO_IS_MASTER` message is multicast and the algorithm is not yet finished). If this message was of the type `WHO_IS_MASTER` the crash detection timer is started and the algorithm is not yet finished. If an `I_AM_MASTER` message has been received a winner exists, which contradicts our assumption. Therefore a process cannot finish the election algorithm without either declaring itself winner, or having received an `I_AM_MASTER` message containing the winner's identification.

5. Current State and Future Trends

We have implemented the Program Service on top of the PORDOS kernel successfully. The Program Service uses the central crash detection approach to determine if one of the servers has failed.

To test and observe the protocols of this service we have developed a graphical animation tool which shows the main protocol events such as election and surveillance messages, server failures, client requests, etc. In addition, the distribution of the code segments on different servers can be observed.

Experiences with the implementation of the Program Service have shown that the surveillance mechanism proposed in this paper is capable of discovering server failures with adjustable reliability and within a predefined time bound. The election algorithm ensures that after a server crash the surveillance is started again immediately.

Since surveillance of processes or servers is an important part of fault tolerance concepts based on replication, we plan for the near future to extract the surveillance

protocols from the Program Service and to provide server surveillance as a separated service for other applications.

References

- (1) Abu-Amara, H. (1988), Fault-Tolerant Distributed Algorithm for Election in Complete Networks, *IEEE Transactions on Computers*, Vol. 37, No. 4, pp. 449-453
- (2) Becker, T. (1990), Achieving High Availability in a Distributed ISDN Control System, Technical Report ZRI 7/90, University of Kaiserslautern
- (3) Birman, K., Joseph, T. (1987), Reliable Communication in the Presence of Faults, *ACM Transactions on Computer Systems*, Vol. 5, No. 1, pp. 47-76
- (4) Bar-Yehuda, R., Kutton, S., Wolfstahl, Y., Yaks, S. (1987), Making Distributed Spanning Tree Algorithms Fault-Resilient, in: Brandenburg, F.J., Vidal-Naquet, G., Wirsing, M. (Ed.), *STACS 1987, Lecture Notes of Computer Science*, Vol. 247, Springer Verlag, pp. 432-444
- (5) Chang, J., Maxemchuk, N.F. (1984), Reliable Broadcast Protocols, *ACM Transactions on Computer Systems*, Vol. 2, No. 3, pp. 251-273
- (6) Cheriton, D., Zwaenepoel, W. (1985), Distributed Process Groups in the V Kernel, *ACM Transactions on Computer Systems*, Vol. 3, No. 2, pp. 77-107
- (7) Cristian, F., Aghili, H., Strong, R., Dolev, D. (1985), Atomic Broadcast: From Simple Message Diffusion to Byzantine Agreement, *15th International Symposium on Fault Tolerant Computing*, pp. 200-206
- (8) Cristian, F. (1988), Agreeing on Who is Present and Who is Absent in a Synchronous Distributed Systems, *Proc. 18th International Symposium on Fault-Tolerant Computing*, pp. 206-211
- (9) Fischer, M., Lynch, N., Paterson, M. (1985), Impossibility of Distributed Consensus with One Faulty Process, *Journal of the ACM*, Vol. 32, No. 2, pp. 374-382
- (10) Garcia-Molina, H. (1982), Election in a Distributed Computing System, *IEEE Transactions on Computers*, Vol. C-31, No. 1, pp. 48-59
- (11) Itai, A., Kutton, S., Wolfstahl, Y., Yaks, S. (1990), Optimal Distributed t-Resilient Election in Complete Networks, *IEEE Transactions on Software Engineering*, Vol. 16, No. 4, pp. 415-420
- (12) Krautkramer, W., Sauer, K., Schlichtharle, D., Knobling, R. (1990), Flexibles Systemkonzept zur Einfuhrung von BK-Diensten im privaten Netzbereich, ntz, (in German)
- (13) Moran, S., Wolfstahl, Y. (1987), Extended Impossibility Results for Asynchronous Complete Networks, *Information Processing Letters*, Vol. 26, pp. 145-151
- (14) Nehmer, J., Becker, T. (1990), A Fault Tolerance Approach for Distributed ISDN Control Systems, 4th ACM SIGOPS European Workshop, Bologna
- (15) Nehmer, J., Gauweiler, T. (1990), Design Rationale for the PORDOS Kernel, Technical Report ZRI13/89, University of Kaiserslautern
- (16) Pohlit, R., Bleichrodt, M. (1989), Dynamically Adaptable Multiservice Switch (DAMS) (project 1059), *Proc. 6th Annual Esprit Conference*, Brussels
- (17) Wybraniec, D., Stamen, F.-J., (1991) Ein zukunfftiges Kommunikationssystem fur den privaten Netzbereich, 21. Jahrestagung der Gesellschaft fur Informatik, to appear (in German)

THE TRANSLATOR'S WORKBENCH AN ENVIRONMENT FOR MULTI-LINGUAL TEXT PROCESSING AND TRANSLATION

M. KUGLER, *M. HÖGE, G. HEYER, R. KESE,
B. v. KLEIST-RETZOW, G. WINKELMANN

TA Triumph-Adler AG
TA Forschung EF
Fürther Straße 212
8500 Nürnberg

*Mercedes-Benz AG
Central Department of Foreign Lan-
guages (ZSD)
Postbox 60 02 02
7000 Stuttgart 60

SUMMARY

The Translator's Workbench provides the user with a set of computer-based tools for speeding up the translation process and facilitate multi-lingual text processing and technical writing. The tools include dictionaries, spelling, grammar, punctuation and style checkers, text processing utilities, remote access to a fully automatic machine translation system and to terminological data bases, an on-line term bank, and a translation memory in an integrated framework covering several European languages. In this paper the architecture of the system and its components are presented. A sample working session is described showing the easy applicability of the workbench to everyday problems in word processing and translation.

1. INTRODUCTION

Due to the increase of international contacts, multi-lingual text processing and translation are getting more and more important. While the costs in computer time and memory are very high for fully automatic translation systems and their performance is still a problem, we have decided to develop an integrated package for assistance, not replacement of translators. The TRANSLATOR'S WORKBENCH, (ESPRIT project 2315) was designed to give assistance to professional translators and secretaries, scientists and engineers working in technical fields in performing multi-lingual text processing and handling large volumes of documentation in one or more languages. At several public demonstrations the public interest in an integrated system providing these facilities was very high.

The Translator's Workbench provides the user with an integrated set of computer-based tools for speeding up the translation process and facilitate multi-lingual text processing and technical writing with respect to sophisticated language checking and help for developing, retrieving, and updating terminology. The tools include dictionaries, spelling, grammar, punctuation and style checkers, text processing utilities, remote access to a fully automatic machine translation system and to terminological data bases, an on-line term bank and term bank building tools, and a translation memory in an integrated framework covering the languages English, German, Spanish, and to some extent Greek. Work is in progress to add the languages French and Italian in order to cover the most important European languages.

2. DESIGN OF THE TRANSLATOR'S WORKBENCH

The overall architecture of the system is shown in Figure 1. The workbench consists of a central management unit and a set of application modules. The manager is responsible for the activation and correct termination of the modules. A uniform communication architecture has been developed to enable easy standardized integration of further components.

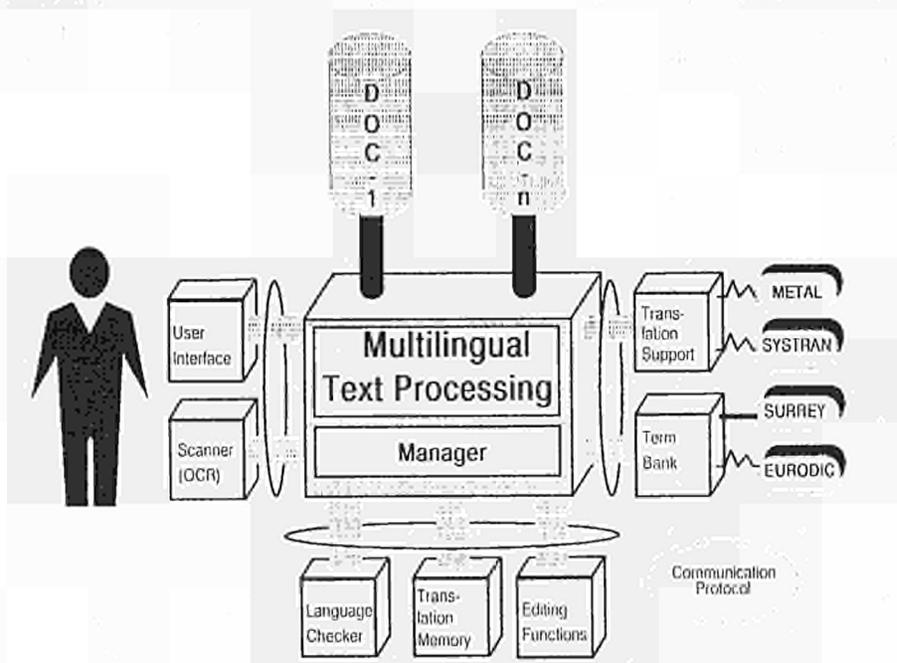


Figure 1: TWB Integration Architecture

3. STATE OF IMPLEMENTATION

After the first two years of the three-year-project, a partial prototype of the overall workbench as well as prototypical realizations of stand-alone modules have been implemented based on FrameMaker and X Windows under UNIX on a SUN 3/80 workstation. Work on a MS Windows prototype with reduced functionality (i.e. language checking and multi-lingual dictionaries) is in progress.

The UNIX prototype integrates the multi-lingual editor FrameMaker, sophisticated language checking facilities, a phrasal translation memory, remote access to the METAL machine translation system and the remote term bank EURODICAUTOM, and an on-line term bank with almost 4000 terms on automotive engineering for each language excluding Greek. The user profile has been tailored to the user's needs, as they were established by a detailed user requirements study. The interchange of documents to and from METAL is format-preserving and adheres to the international ISO standard for ODA/ODIF (ISO 8613) (1).

4. USER REQUIREMENTS

In the first phase of the software lifecycle, an extensive study of user requirements was performed by the Mercedes-Benz AG and the University of Surrey. It included a questionnaire survey of over 100 professional translators within Europe, close observation of six translators at work and in-depth interviews with ten translators.

The outcome of this user requirements' investigation "showed that there is a need for computer-based tools in professional translation circles. These tools include 'smart' editors, terminology data banks, and remote access to machine translation systems. The 'smart' editor will provide conventional word processing facilities, support the analysis of documents in both the source and target languages, and incorporate syntactic and stylistic analysis tools." (2)

5. THE TOOLS

5.1. Remote Access Facilities

Access to external term banks usually is a very tedious task. The TWB access module makes term bank interrogation more user friendly and provides two major advantages: the user does not have to know anything about networks and he or she does not need an interrogation language to receive information. At present, an access module to EURODICAUTOM, the official CEC terminology database has been implemented by the Universidad Politecnica de Catalunya and SNI CDS.

5.2 Language Checking

Conventional word-based spell checkers are available for all the languages of the project, two of them newly developed by SNI CDS (Spanish) and L-Cube (Greek). A new feature by Triumph-Adler is the context-sensitive spell checker for German exceptional cases, many of which can neither be handled by a word-based spelling checker nor by a grammar checker. This extended spell check treats special capitalization and concatenation problems (e.g. "in bezug auf" vs. "mit Bezug auf", "radfahren" vs. "Auto fahren"). German and Spanish grammar checking and style checking is done by accessing the parser of the automatic machine translation system METAL (SNI, SNI CDS). For the PC version, an ATN-based phrase parser with heuristics on sentence-level discovers errors in agreement (case, person, number, gender within phrases; number between subject and verb phrase)(3) (4).

5.3. Term Bank and Term Bank Building Tools

The corpus-based Machine Assisted Terminology Elicitation (MATE) toolkit developed by the University of Surrey (5) enables translators and terminologists to access term banks, text corpora and facilities for customized dictionary production, and thus enables the translator to store, retrieve and update terms. The term bank developed by the Universities of Surrey and Heidelberg within the project currently contains almost 4000 terms in the domain of automotive engineering in the languages German, English, and Spanish (6).

The term bank available covers the whole range from definition, grammar, usage, collocation, and equivalents of a term in question up to encyclopedia, hierarchy and word family.

5.4. Translation Memory

Due to the vast knowledge necessary to do a full-fledged automatic translation, only few systems exist that are open to a larger market. None of them can be used on a small machine with little memory and disk resources. The more primitive dictionary-systems that do simple word-to-word translation are not applicable in most cases due to the low quality of their output when translating text. Therefore, while providing remote access facilities to the automatic translation system METAL developed by Siemens, we have followed the intermediary approach of providing partial translation as an additional tool which can be used on personal computers. The trainable translation memory developed by the Fraunhofer Gesellschaft (7) enables the user to quickly retrieve previously stored translations in a flexible way, i.e. the program tries to find the largest known source language sequence and supplies the corresponding translation. Thus the translation memory provides a fast way of reproducing translations of variable length, giving the user the opportunity of interactively supplying the translation for unknown words.

These features make the tool well suited for formalized and repetitive text types, e.g. technical texts and business correspondence. New manuals often consist of some new text blocks mixed with old text, which had been translated before, business letters contain highly standardized phrases and expressions.

Furthermore, when writing manuals etc., it is important to ensure that a technical term is translated the same way anywhere in the text, in order to provide a uniform and standardized surface. Thus a translation memory can provide more uniformity to a task involving several people in translating one text.

5.4.1. How it Works

The translation memory is based primarily on statistical methods, it works with Markov models of trigrams. Word S3 in the sequence S1 S2 S3 is translated to T3 in the sequence T1 T2 T3 (S: source language, T: target language). Frequency information is stored in the database, so if several translations are available, the more frequent one is selected.

Prior to the translation a training phase has to be performed. This is done in the same text processing windows as the translation.

In the training phase, a variable number of words in language A can be linked to a variable number of words in language B. The system asks the user whether to update the dictionaries with the unknown words and then presents two windows with symbolic links between words or word groups of the two languages. Only translations that have not been trained previously are asked for, the others are suggested automatically and have to be confirmed by the user, thus speeding up the training process.

Whenever a sequence of words turns up in a new text, the corresponding translation is retrieved. The larger the sequence matching the originally trained text, the better the grammatical structure of the target language can be reconstructed. If the text is very dissimilar from previously trained material, the worst case would be word-to-word translation with occasional untranslated words, depending on the size of the dictionary. This is obviously not desirable, so the best application for this tool is text with repetitive expressions.

The method is inherently language independent, thus the database can easily be trained for any pair of languages.

5.4.2. *The Database*

The performance of the system with untrained text did improve only slightly by using a large non-specialized dictionary (25000 word forms, German/English). Translating text in the range of computer manuals showed that the vocabulary is highly application specific. Therefore work concentrates on providing a technical dictionary (technical terms, economics) on the one hand, and training of phrases from business correspondence on the other. The best performance was reached when training the system on the first paragraphs of a new manual and then applying it on the subsequent translation of the remaining text.

As the number of possible sentence constructions is enormous, training of full sentences requires too much time and provides too little output. The 'phrase unit', however, seems to be easier to standardize and thus looks the appropriate size for storing translations. We are at present experimenting with easier decomposition of sentences into trainable phrases.

6. Example of a Working Session

Using the language checker, the source text is prepared for translation by detecting ill-formed input and complicated constructions. Having corrected the source text, unknown terms can be looked up in the term bank, which among others provides encyclopedic knowledge, the translation(s) of the term and information on its usage. If a term is not available from the on-line term bank, a link to a remote term bank (e.g. EURODICAUTOM) can be made and the information can be retrieved from there.

The text is then translated, which can be done either interactively using the statistical translation memory or in batch mode by sending the whole text or parts of it to METAL, the automatic machine translation system.

The translated text can then be post-processed using the language checker of the target language. Source and target text can be viewed simultaneously in order to enable easy comparison and control of the original and the translation.

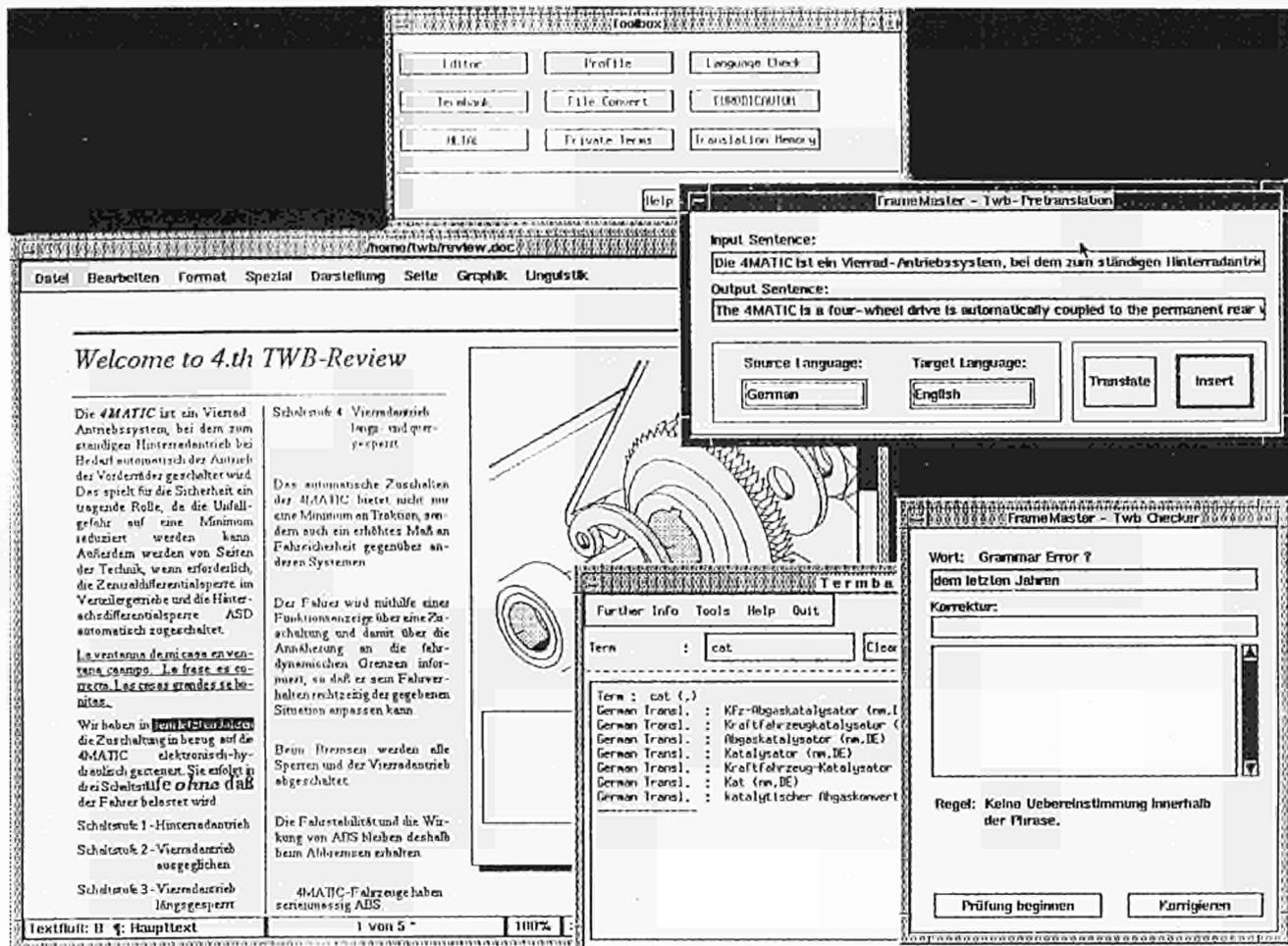
When using the translation memory, part of the revised translation could be trained immediately to be available for future use.

All the tools mentioned are integrated into a common system with a unified user interface. Most of the tools can be called from within the editor, thus enabling the user to perform the tasks without leaving the editing environment. Formatting information is not destroyed by applying most of the tools, when doing the batch translation by METAL, format information is preserved using the ODA/ODIF conversion format.

At present the domain of application is in manufacturing, especially within the automobile industry.

Please see for an example the following figure, showing a screendump of the UNIX prototype. Of course, when using the system, not all windows are kept open all the time. When necessary, each window can be minimized or restored with one or two mouse clicks.

Figure 2: TWB screen - work in progress



7. Performance Evaluation

To guarantee the development of not only user friendly but also adequate translation software, the focus of TWB was put on user participation throughout the different phases of the software lifecycle. The Mercedes-Benz AG with their large translation department took over the job of requirements definition, software testing, and evaluation.

- The user point of view in software development comprises two aspects: on the one hand the software has to offer a certain quantity of functions (see Checklist in 8) and on the other hand the functions available have to meet a high quality standard.

7.1. Quality Factors and Evaluation Metrics

For the translators as the main target group it was found that user satisfaction mainly involves the quality factors reliability, usability, and efficiency. Having roughly defined the quality factors relevant for translators, the corresponding criteria had to be determined and measurable quantities found. Each criterion can be measured and evaluated on three levels, i.e. the functional level, the interface level and the content level (for tools offering the translator a certain informational output, e.g. term bank, checker, translation memory...). Figure 3 (next page) shows how the different quality factors can be measured quantitatively on the three different levels.

The testing of TWB involves two different testing modes, i.e. long term testing and scenario testing. Testing and evaluating the test results has to take into account the development status of the software, the type of the software and the test scenario. It turned out that it is "only after the questionnaires and checklists are prepared for the specific scenario test that both users and developers may define exact target values for the individual measurable quantities" (8).

At this stage of the TWB project, two scenario tests were performed, providing the developing teams with useful clues with regard to the functional, interface and content problems their software modules evoke at this stage of development when being used by translators. The constructive criticism made by translators and the evaluation team will guarantee that the outcome of TWB reaches as high a quality standard as possible within the scope of the TWB project.

7.2. Why Use a Translator's Workbench - Cost vs. Benefit

A detailed cost-benefit analysis for the translator's workbench compared to traditional manual translation is not yet possible at this stage of the project. However, translators' general reactions to the new facilities were positive. Furthermore, the workbench only offers the various tools, it does not force the translators to use them. We therefore expect each user to utilize whatever is agreeable to him or her.

Even without explicit guidelines, though, the reusability of translations and quicker access to terminology should lead to a better standardization of terminology and easier dissemination of information.

Due to the wider distribution of text processing facilities many translators nowadays have access to a computer. Mere manual/typewritten translation work is getting rare. It is therefore only reasonable to have means of help adequate to the wider possibilities a workstation or PC offers to the user.

Figure 3: A user oriented software quality model

QUALITY FACTOR	CRITERIA	MEASURED QUANTITY		
		functional level	interface level	content level
reliability	task adequacy	checklist items	WYSIWYG	ratio of searched/found terms actual/detected errors searched/found information cat.
	correctness	number of failures mean time to failure time when failure occurs number of cumulative failures failure type		suitability of presented information for specific purpose correct/incorrect data output
	error tolerance	undo facility escape function error messages		
usability	ease of learning	time needed for training program frequency of help/documentation use time spend using help/documentation time needed to achieve a performance criterion number of error messages		comprehensibility of output texts (definitions etc.)
	ease of use	time needed for task	actual/minimal number of keystrokes	
	task relevance	actual/maximal number of function usage availability of help facility/documentation success/failure of completing a given task		
	comprehensibility	understandability of help facility documentation system messages	clarity of layout mnemonic labels	
	consistency	similarity in performance	consistent layout	
efficiency	execution efficiency	response time for queries response time for batch programs	time needed with button/ menu version	
	performance efficiency	storage space needed amount of text translated in a given time compared to conventional methods success/failure of completing a given task in a fixed amount of time		

8. Future Objectives

8.1. Products

SNI is interested in a workstation version of the Translator's Workbench integrating several tools using the FrameMaker environment (under UNIX), while TA/Olivetti will launch a product aimed at the MS-DOS PC market, containing an editor, conversion to and from the main text processing systems, a translation memory, and language checking integrated under MS Windows, covering the languages English, German, French, Italian and Spanish.

8.2. Perspectives - What to do Next

In the course of the project, a second line of development has emerged with the aim of porting part of the workbench to the MS Windows environment. Although at the present stage a comparatively small number of modules is available, several partners are working on PC versions and we expect to present almost full functionality within short time.

Focus now has to be on the evaluation of the different tools and on unification of the knowledge bases, (i.e. dictionaries) within and across languages.

9. Literature

- (1) DELGADO, J., JORDAN, F. and MEDINA, M. (1990). Access to automatic translation machines using X.400 messaging and ODA documents. Proceedings of the 10th International Conference on Computer Communication, 5-9 November 1990, New Delhi (India).
- (2) FULFORD, H., HÖGE, M. and AHMAD, K. (1990), User Requirements Study. Mercedes-Benz AG, University of Surrey 1990
- (3) THURMAIR, G. (1990). Parsing for Grammar and Style Checking. COLING 1990.
- (4) WINKELMANN, G. (1990). Semiautomatic Interactive Multilingual Style Analysis. COLING 1990.
- (5) AHMAD, K., DAVIES, A. et al. (1990). A Methodology for Building Multilingual Termbases and Special-Purpose Lexica. TWB Technical Report, University of Surrey
- (6) ALBL, M., KOHN, K., POOTH, S. and ZABEL, Zabel (1990). Specification of Terminological Knowledge for Translation Purposes. TWB Technical Report, Universität Heidelberg, Institut für Dolmetschen und Übersetzen
- (7) KECK, B. (1991). Translation Memory - A Translation Aid System Based on Statistical Methods. TWB Technical Report Fraunhofer Institute IAO Stuttgart
- (8) HÖGE, M., WIEDENMANN, O. & KROUPA, E. (1991). Evaluation of the TWB. TWB Technical Report, Mercedes-Benz AG

On Mobile Work and Elusive Offices: The Elusive Office Project

Werner B. Korte, Simon Robinson, Norbert Kordey, Kerstin Keil
empirica
Gesellschaft für Kommunikations- und
Technologieforschung mbH
Kaiserstraße 37
D-5300 Bonn 1

Peter Heuschötter, Reinhard Nachtsheim
CLS Computer Lern Systeme GmbH
Kaiserstraße 31
D-5300 Bonn 1

Summary

The objective of the Elusive Office (ELO) project is to develop an ELO-System (ELOS) which will allow mobile workers to carry out their tasks, regardless of their location in the same effective manner. An ELOS will provide transparent access to such services as communications, present an integrated view of applications using a high-quality graphical user interface and support the user by means of a computer learning system. Software will be based on state-of-the-art techniques. The ELOS hardware will be a modular, but closely integrated, system based upon a laptop computer with portable cellular telephone, portable fax and portable peripheral input / output devices such as printers, scanners. This hardware and software combination will provide effective office functionality with a sophisticated communications capability. One major aspect of the project is the provision of adequate support and of a high quality interface for the end user. The current plans are for a graphical user interface driven by a pointing device, together with a keyboard for character input. Later in the project an assessment of pen-driven devices is planned.

The paper starts with some background information on mobile working and gives an introduction to the ELO project. The major part of the paper describes work on applications investigations in ELO, the requirements capture, analysis and specification (undertaken by empirica) and the ELO user support system with integrated learning modules (the computer learning system developed by CLS). The concluding sections give an outline of the ELO- System (detailing the selected underlying software, hardware, communications and (generic and specific) software being developed by the Consortium) and the achievements of the project to date.

Introduction

This paper has been jointly written by two of the partners on the ESPRIT Project Elusive Office (ELO) - empirica Gesellschaft für Kommunikations- und Technologieforschung mbH and CLS Computer Lern Systeme GmbH. The major foci of the paper are:

- investigation of potential applications and user groups,
- development of detailed user requirements and

- a support system - the computer learning system (cls) - targeted at the special problems of mobile workers.

The first two activities are important for ELO as truly mobile working is a (relatively) new area: it is more than just a simple extension of the simple tele-working that has been a major feature of some sectors of the UK IT industry for some years. Analysis of user requirements led to the computer learning system (cls): one of the novel features of the ELO-System which is being developed specifically to address the demanding support requirements of a mobile worker.

- The paper starts with an overview of the concept of a mobile office, putting it in a technological and market context, and looking to the future. It became clear early on in the project that there a number of critical features for a successful ELO-System: such as integration, the need for careful identification of user requirements and the central need for adequate support for workers isolated from their colleagues or from the support services of the company's head office. There is then a brief description of the aims of the ELO Project, viewed against this background.

The next sections of the paper deal with the applications investigations - using analysis of occupations, questionnaire techniques and more detailed case studies against a knowledge of the available and emerging technologies in IT&T - producing an overview of potential ELOS users and identifying the two specific trigger applications of the "Presenter" and the "Macher(in)" (High Power Executive). This leads to a discussion of the development of specific user requirements and the adoption of the "work process" approach. Although specific applications will be used for the 1992 Demonstrator, a considerable effort is being made to generalize to the wider application area identified in the initial occupational studies. The critical aspect of support for the ELO-Worker is then addressed. Not only does the "frame approach" of the computer learning system (cls) provide help and learning assistance for the remote worker, it also forms the mechanism for driving the system in accordance with the work processes identified by the user requirements exercise. The paper concludes with details of achievements to date.

2. What is Meant by a Mobile Office?

A mobile office could be described simply as mobile work which is supported by mobile information technology and mobile communications - but this may be a somewhat simplistic definition.

Most people probably understand "mobile office" to mean a conventional office put on wheels of some kind: the office is transferred to a car, to a boat or even has wings.

But this is only one view: a "vehicle perspective" - which might include the car-office, the train-office and so forth. In an example from AEG, all the contents of a typical, conventional, building-based office (including telephone, fax and even the secretary!) were put on wheels. A radio network provided the necessary connectivity. Note that this particular variant of the vehicle perspective on the mobile office needs a big vehicle. The vehicle's role is to provide not just personal mobility but also space, a source of power, and the ability to carry the heavy items of office equipment.

3. Moving away from the Vehicle Perspective

However, the very trends have led to the ability to provide mobile integrated communications and information processing power are now removing the need to

install equipment in a vehicle. We nearly have the technologies to allow us to take the functionality, the fax, the telephone - maybe even some features of the good secretary - and carry them around on foot. As equipment becomes increasingly portable, so the mobile office begins to take on a new meaning.

Looking at portable, personal computers, battery life can for some machines now be measured in days rather than minutes or hours. AT-class computers no longer require weight lifters: these days powerful computers come not just as "luggable" laptops, but in pocket size even as palm-top systems. Though whether it is really desirable to be able to balance your computer on your hand is another question! As information processing systems are becoming more portable, so public networks suitable for communications from almost anywhere are becoming available.

The technological push and non-technological trends discussed below are leading to a surge in demand for information processing and communications support for mobile work. The shipment of laptop PCs to the European market (not forgetting some indigenous production) has nearly doubled in the last year.

Turning now to the communications part of the equation, the uptake of cellular telephony has far exceeded expectations. Germany lags the UK and the US in cellular coverage, but even in Germany the current, C-net is sufficiently popular to cause network congestion problems. The next wave, the D-network-based on the pan-European standard Group Special Mobile, offers cross-border compatibility, digital transmission and much higher capacity, and will help reduce current congestion as well as other limitations. However, the Bundespost was optimistic in announcing its introduction in 1990. The analogue C-network is also getting a new lease of life in the former German Democratic Republic, following unification.

This scenario opens up a number of opportunities and pitfalls. These technologies are being taken up separately. As a result, portable computers run software designed for stand-alone PCs, even though the trend is towards networked and distributed computing. There is also emphasis on multi-processing operating systems and GUIs. The connectivity of high-speed LANs helps integrate software systems in organisations, but these networks omit mobile computing. It is still a challenge to provide effective integration of mobile communications media with portable computing.

4. Non-Technological Triggers for Mobile Work

It is not just these technology trends which are important considerations in supporting mobile work. There are also trends towards more and different kinds of mobility at work, and changes in the character of the work itself. Organisations are becoming increasingly decentralised and there is also a shift of emphasis in organisations from production efficiency to quality of customer service. Both these trends are reflected in increased frequency of contact with clients and client-based work, leading to greater mobility in the work environment. The way in which work is done has also been changing. There is increasing flexibility in work timing and place, and a tendency to mix work and leisure.

5. The Evidence in the Market Place

Altogether, there is now more pressure towards mobility and/or doing work whilst being mobile. What we need to understand is what is driving mobility in work and how it can be quantified. Who is likely to become a mobile worker? For what proportion of their working time? Is this trend likely to change?

Given the benefits that can be accrued from the trend towards decentralisation in organisations (e.g. closeness to product markets, unit flexibility, closeness to labour markets), workers' increased interest in combining leisure and work and varying the place and time when work is done, coupled with the widespread undersupply of qualified personnel, it is likely that work will take place when there is an opportunity to work. The trend towards mobile working is thus likely to continue, whether we like it or not.

6. A Problem of Integration and the Role of ELO

Anticipation of trends in technology and organisational change led to the formulation of the ELO - Elusive Office - project. ELO builds upon the enabling technologies of cellular telephone networks and portable computing, and through four main work areas involving

- applications investigation and requirements analysis
- hardware, software and communications integration and development
- state-of-the-art computing techniques (object-oriented programming, GUIs, etc.)
- computer-based learning, and
- security.

The project aims to be able to demonstrate prototype results in 1992, with a 1995 target for the marketing of products for the "elusive office".

7. ELO: The Elusive Office

The main objective of ELO is to develop an ELO system (ELOS) supporting the maximisation of a workers mobility, i.e. their flexibility in work location and working time, allowing the mobile worker to work in an effective manner and in the same way regardless of location.

Thus the development of:

- an architecture, philosophy and verbal descriptions of functionalities for the development of ELO type applications together with
- an object-oriented software toolkit and the associated application framework for easy generation of integrated applications, with special consideration of security requirements, and an effective learning system

will constitute a major component of the ELOS to be developed.

It will provide:

- transparent access to services, such as communications,
- full integration of applications, present consistent user view of applications
- other features becoming the norm in software development (multi-processing, a quality graphical user interface, and so forth),
- integration of software items (both applications and the computer-based learning support system).

In addition appropriate hardware with sophisticated communications capabilities, i.e. an integrated modular combination of laptop, portable cellular phone, portable fax and

portable peripheral I/O devices such as printers, scanners, etc., will be created. The current plans are for a graphical user interface driven by a pointing device, together with a keyboard for character input. Later in the project, an assessment is planned for pen-driven devices. These are now becoming available and have a number of attractions for certain user groups: offering good sketching performance as well as limited handwriting recognition.

The project does not intend to develop most of these hardware components. Readily available components will be employed in the assembly of a Demonstrator in 1992. However, the project intends to develop specifications for components which are not available or need to be enhanced to achieve a full ELOS.

The work in ELO is application-driven, i.e. potential application areas and applications are being identified systematically and researched in depth to determine current and future requirements. Demonstration of an integrated ELOS in real working environments is planned within the project. Such demonstrators require the development of application software specific to a particular (ELO) application.

The project is structured into technology and applications "strands". The applications work (which covers investigation of demand factors leading to requirements identification) is used as the basis for system development and integration work. This feeds into the implementation and evaluation of a demonstrator system.

empirica (D) is the prime contractors in the ELO project and responsible for applications and requirements capture, analysis and specification as well as the development of supportware for easing the use and introduction of an Elusive Office System (ELOS) in work environments. The Rutherford Appleton Laboratory (UK) provides the project with an architecture and framework for software and communications integration and is developing the generic ELOS applications. Alcatel-SEL (D) provides the integration of hardware and communications. CLS Computer Lern Systeme GmbH (D) provides the computer learning support systems which will be integrated as far as possible with the applications and is developing the specific application software. Fraunhofer ISI (D) is responsible for testing and evaluation of the demonstrators. Associated partners (Realace - IRL, ÖVA Versicherungen - D -) and sub-contractors (OnStream Resources Ltd - UK -, Intrepid - IRL) cover user-interface design work and security, as well as offering the project its test-bed infrastructure.

8. Applications Investigations in ELO

ELO aims to provide a systematic approach to the problems associated with mobile work. It began by considering the range of occupations available, to try to identify typical mobile workers. There are no statistics on levels of mobility in occupations, so the ELO project team used expert sources to judge the occurrence of mobility in various occupations. We also took on board fourth-sector approaches to systematising economic activity: the identification of an "information sector" which is associated with information occupations. This work goes back to Porat in the 1970s (9). Finally, we looked in more detail at occupations currently using mobile forms of information and communication technology.

Seventy-nine case studies of occupation and a questionnaire survey in 214 European organisations were carried out, leading to the identification of the following ELO user types by empirica:

- Sedentary
- Solo-worker

- Teamworker
- Delegater
- Macher(in) (High Power Executive)
- Traditionalist
- Presenter
- Visitor
- Engineer
- Driver.

These groupings do not correspond in a simple, one-to-one way to particular occupations. Rather, they take into account both job and individual characteristics. For example, the "Driver" group includes emergency staff. The intention was not only to systematise the area and get some kind of insight into a variety of ways of working, and the associated requirements for communication and information processing. We also wanted to identify so-called "trigger groups", which would allow us to identify "trigger applications": those with the most immediate product potential, and capable of being selected for implementation and demonstration by 1992.

Clustering of occupations into three different "bands" of ELO trigger potential was done by empirica using the following criteria:

- mobility
- type of activity
- communications requirements
- use/creation of information

This produced the classification outlined in Figure 1.

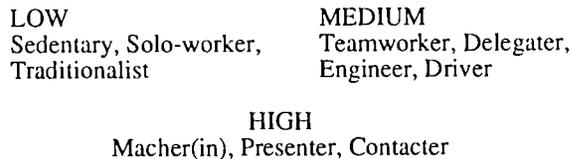


Figure 1: ELO Trigger Potential

9. The Potential ELO Users

Some ELO types clearly have no trigger potential. For example, the "sedentary" type contradicts the assumption of mobility because these are people in occupations where, even if there is pressure towards mobility, they have successfully managed to resist it, or the division of labour in their organisation allows them to avoid mobility. Similarly with the "traditionalist": though their the work would provide efficiency improvements through the use of information technology, they adhere to traditional ways of working - delegating information and communications tasks.

Identification of trigger groups used similar dimensions, particularly

- the size of the group
- its position in the organisation
- the level of competition prevailing

- the group's current use of MICT (Mobile Information and Communications Technologies), or its immediate interest in these.

The following two ELO "trigger groups" were identified:

- High Power Executive ('Macher(in)')
This group includes
 - managers
 - directors of small businesses
 - heads of department in medium to large organisations
 - scientific staff and engineers.
 Their market size is about 300,000 (previous FRG)
- Presenters / Contacters
This group includes sales people with
 - customer types ranging from private individuals to medium sized enterprises and
 - products ranging from insurance policies to delivery vehicles.
 Their market size is about 600,000 (previous FRG).

10. ELO Users' Requirements

The next step was to investigate the requirements of these two trigger groups in terms of the specification of the 1992 ELO Demonstrator. The aim was to capture, analyse and specify requirements for a system supporting a particular segment of work of the chosen ELO trigger groups (Presenter and "Macher") centering on an overlap of functionality between these two user groups which could clearly show the pillars of the ELO project. "Phone Management" was taken as the initial focus for requirements analysis. Using this as a core functionality other activities could be added in a task-centred fashion. As will be shown subsequently this was gradually expanded to encompass all the activity planning, communication tasks and system support for the execution of work processes exemplified by the particular work processes of one target user group.

10.1 "Phone Management": Telephone Communication Requirements

The methodology used by empirica to generate ELO users' requirements for telephone communication focused on the tasks these people perform, but also took account of technological trends. A parallel process of review of current systems, and investigation and analysis of each group's typical tasks, led to a process of synthesis, where user needs and technological parameters are integrated and modelled. This is then concretised in prototypes which were subjected to an iterative cycle of evaluation and validation. The final deliverables were then produced.

As the project is concentrating on "case-carriers" in ELO, not on vehicle-based systems the ELO system has to provide all the functionality of today's portable systems, plus the added value of the telephone (both from its integration into the ELO system, and the software control of telephony afforded by ELO). Thus ELO will make work independent of location.

Depending on the ELO type, comprehensive support may include quite diverse application functionalities. There are, however, some commonalities of functional requirements, one of which is a need for voice telephony. Voice telephony cuts across ELO types and tasks, and is thus a good example of the application of the Requirements

Generation Methodology. As part of our methodology we looked at current systems for voice telephony and re-examined their functionality for their integration potential. Moreover, we examined current software products which "can" be used for voice communications as well and identified some basic functionalities that need to be provided at the level of communications support software in an ELO system.

Our methodology also involves looking at users' tasks, so that both can be used to generate the final requirements list. Through observation and task analysis, we looked in detail at the activities associated with using the telephone, and built up a catalogue of the tasks which users perform either in preparation for, during, or after a call. This modelling exercise showed how complex even seemingly simple tasks really are. The analysis produced a vast flow diagram, only about one third of which was concerned with the actual voice conversation.

10.2 Requirements for Activity Planning and Communication

In the course of the project it became apparent that there exist a number of activities which are closely related to the "telephone call" activity. These include:

- preparing for capturing incoming telephone messages when out of the office
- receiving such telephone messages upon return to the office
- note-taking during and/or after a telephone conversation
- using other media (e.g. documents, PC) in parallel with the telephone call
- getting other people to the phone
- involving other people in a telephone conversation.

Accordingly it was decided to expand the requirements analysis and to analyse and generate requirements for activity planning and communication. In order to give the reader a quick overview of the areas for which requirements were specified as tasks in the "Activity Planning and Communication" task catalogue a list of the nine design domains for which tasks were specified is given. Further details and a complete list and description of these tasks can be obtained from the corresponding empirica report (6). The design domains are as follows:

- ELO-Mobile Unit
- locations
- people
- information capture and display
- dialogue and telephony
- messaging
- diary
- user task handling
- integration features.

10.3 Requirements for Work Process Support

Moreover, empirica identified the need for support of work processes where a number of geographically distributed actors with different roles (e.g. insurance agent, Head Office clerk, client, information provider) are working together. This led to the decision to specify requirements for work process support for claim handling: a frequent (i.e. daily) activity. In order to give the reader a quick overview of the areas for which requirements were specified in the "Claim Handling" task catalogue a list of the so-called

section tasks is given. The tasks specified under these headings cover the entire process of claim handling involving various actors at different locations. Further details and a complete list and description of the tasks specified for each section task can be obtained from the corresponding empirica report (7). The section tasks are as follows:

- Claim Notification on Answerphone
- Claim Notification on Telephone
- Claim Handling
- Claim Form Inquiry
- Claim Form Completion and Print 1
- Claim assessment By Assessor Initiated By Agent
- Client Visit Preparation and Claim Form Print 2
- Client Claim Visit
- Receive Back Mailed Claim Form
- Post Visit Actions.

10.4 Format Used for the Specification of Requirements

Different means for specifying and presenting the requirements were used by empirica. These include the generation of task catalogues, screen examples, and process diagrams.

10.4.1 Task Catalogues

The requirements capture and analysis work included the capture of status quo information on relevant tasks and work processes. The information was then projected into a future way of working and into a particular technology scenario ("creative leap") to develop requirement specifications which were partially already validated in discussions with users.

All requirements - those for activity planning and communication and those for the support of the work process of claim handling - were specified by empirica using the format of a task catalogue. It should be noted that the task catalogue also contains user interface recommendations, e.g. several task aspects can be presented as icons. The format of the task catalogue entries is as follows:

"Task"

< Task name >

"Objectives"

< text of user objective, containing the user requirements, where:

- "-" represents a goal which is to be met directly
- "--" represents a goal which is to be met indirectly >

"Notes"

< any clarifying text interceding between "Objectives" and "Process"; optional; deals with:

- conditions under which a task might be carried out
- implementation options and choices >

"Process"-Section

<description of how the task is executed by the user; it implicitly includes a presentation of a Task Form to the user as described in the "Task Form"-Section >

"Icons"-Section

<description of the task icon; only for User Tasks; optional >

"Task Form"-Section

<description of what the user is presented with when initiating a particular task >

e.g. StandardTaskForm: format:

- **"Task Form Label"**

"Info": Infofield:

allows assignment of a note directly to an active task avoiding the need for opening an additional note

" ": Transformfield

allows changing the Task Form into a note, message, etc. without losing specifications like addressee/callee

" ": Kontextfield

automatically created when the user starts a User Task; contains identification information: "Kontext" information related to a task, e.g. "Scha densbearbeitung", "Betrifft" information related to a person

- **Buttons:**

"Delete": deletes the current task form and/or changes the task status

"Shrink": shrinks the current task form to an icon and/or changes the task status

"Entstehungsgeschichte des aktuellen Task": invokes TaskOriginExplore

- **"Weiter Mit"-Section**

< allows the user to select the next task out of a list of tasks which is updated depending on his/her current work process context (note: most ELOS tasks are permanently available via the Main Menu Icon Bar) >

- **Fields in Task Forms**

these encompass date field, time fields (time field, relative time field, time period field), person fields, money field, icon field, contract field, role abbreviation field,

selection list field, warning fields, context field, task title field and a task subtitle field.

10.4.2 Screen Examples

Screen examples were also developed by empirica for particular tasks of the "Activity Planning and Communication" task catalogue. These illustrate how an ELOS screen might appear to an ELOS user in different situations. Figure 2 gives an illustration of a typical screen providing the user with the Main Icon Menu Bar, figure 3 illustrates a screen which the user will see when preparing for a telephone call to a particular person. Both are intended to be indicative rather than in any sense definitive.

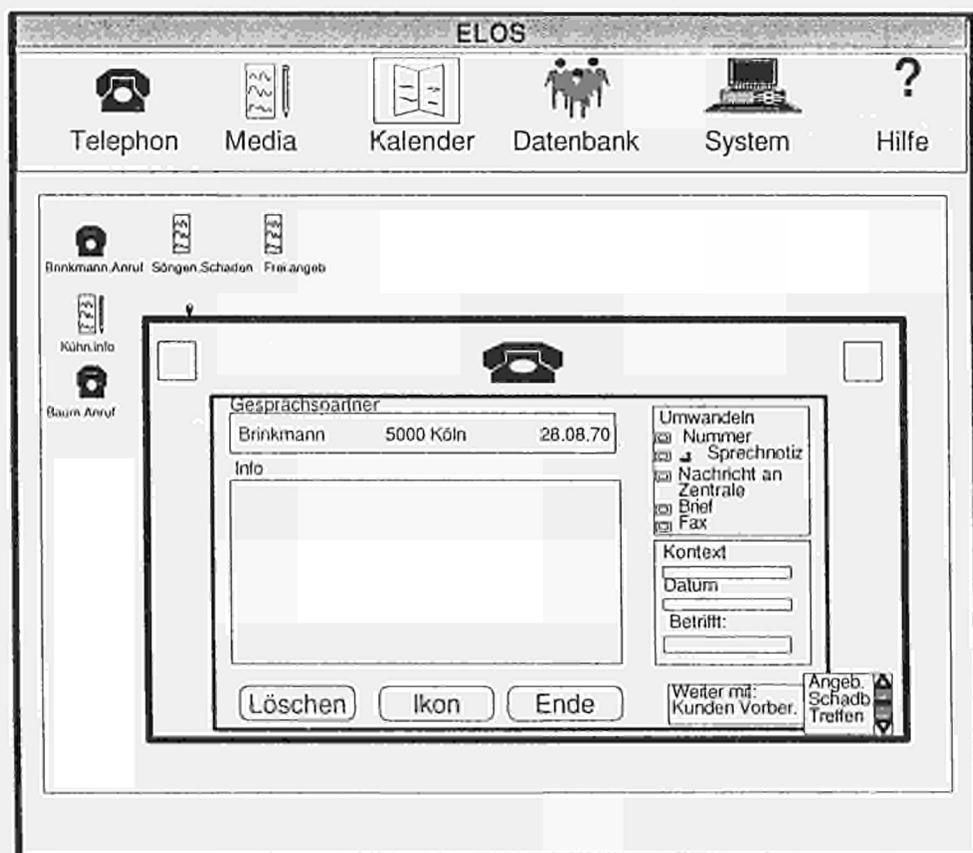


Figure 2: ELOS Screen Example: Typical Start Screen


Anruf - Vorbereitung

Gesprächspartner

Name	Vorname	Wohnort	Geburtsdat.
Br			

Umwandeln

- Nummer/Person
- Sprachnotiz
- Mitteilung an Zentrale
- Letter
- Fax

Kontext

Info

Söngen
 Ulrich
 Werther

 Steffan
 BD
 Sparkasse
 Alice
 Busch
 Auto

Kunden →	Brand	Wilhelm	5020 Roden	25.10.54
Zentrale →	Braun	Lotte	5180 Secht	2.11.64
Privat →	Brinkmann	Hans	5000 Köln	28.08.70
	Brix	Emma	5300 Bonn	19.04.38

Löschen

Ikon

Ende

Weiter mit:

Figure 3: ELOS Screen Example: Telephone Call Preparation Screen

10.4.3 Process Diagrams

For each section task of the work process "claim handling", process diagrams were developed by empirica. These process diagrams illustrate the sequence and interdependency of the tasks. The following figure presents the example of the section task "Claim handling".

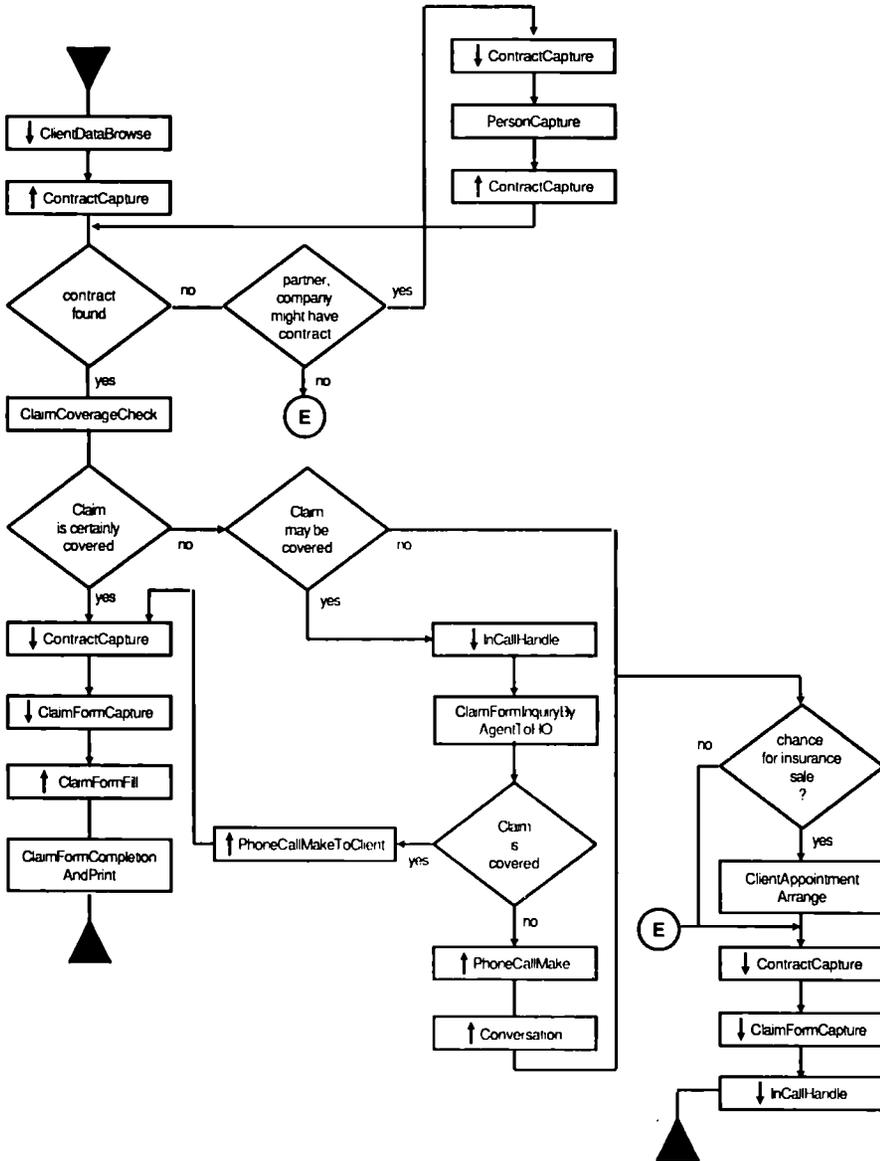


Figure 4: Process Diagram: Section Task "Claim handling"

11. Characteristics of the ELO-System and Motivation for the **cls**

Although the explicit user requirements can be derived from the ELO worker's (ELOW) application domain, they do not reflect the implicit requirements for user support, which emerge from the specific characteristics of the ELOS:

- (a) mobility,
- (b) integration of software and hardware-components to satisfy pre-defined application dependent requirements and
- (c) provision of an "open system" into which other tools can be easily integrated

These characteristics supplied the motivation for the provision of a user support system - the computer learning system (**cls**) - as a primary component of ELO-Systems.

11.1 Mobility

The mobile character of an ELO-System produces a major impact upon the potential ELO-Workers and their accustomed work habits. An ELO-Worker will probably be representing their companies and using computer-supported work methods on their own, remote from their head-office. That is, without being backed-up by the support and expertise of other colleagues or company service departments. Although the ELO-System (a lap-top like computer system with domain-specific software support) will be the ELO-Worker's standard equipment, it is not assumed that ELO-Workers are, nor should they necessarily become, computer experts with sophisticated software and hardware knowledge. The ELO-Worker's professional education and background, and their focus of attention is directed towards their particular domain knowledge. Their tool, the ELO-System, serves only as a means of achieving their work-oriented goals. For these reasons, there is a clear requirement for a support system which helps the ELO-Worker to compensate for the lack of various highly specialized technical skills that would be necessary to operate a 'raw' system effectively. In other words, the ELO-system itself should be transparent to the ELO-Worker, and tailored both to the individual and to the work-domain.

11.2 Integration of Heterogeneous Services

The ELO-System integrates communication services, (such as voice-communication, fax and data-communication) and various (generic and domain-specific) applications into a complex system. The resulting level of complexity requires an architecture and a user support system which provides the ELO-Worker with a (relatively) simple and "work-process" oriented way of viewing and operating the system.

11.3 Integration of Various Tools into the ELOS and into the **cls**

A number of distinctions, and associated points concerning integration, can be made concerning the tools available within an ELO-System and the interfaces of these tools both to the user and to the **cls**:

- ELO and non-ELO tools,
- user interface of ELO and non-ELO tools,
- **cls** interface and integration with ELO and non-ELO tools.

The remainder of this section attempts to indicate what is currently planned and to give some indication of what may be possible in the future.

An ELO-Tool is a piece of software specifically written (or modified) to conform to the object-oriented Framework of the ELO-System. It is therefore able to make use of the common presentation systems and techniques for access to other tools and facilities of the ELOS. Non-ELO tools (such as third-party word-processing systems, spread-sheets ... for which there is no access to source-code) can only be integrated in a limited fashion: to some extent they must be treated as "black boxes".

The **cls**, the ELO Framework and its related components will provide a platform for integrating third party products (principally non-ELO tools at present) in a comparatively easy fashion. But, as stated above, the level of integration and the extent to which those non-ELO tools can benefit from the underlying architecture, especially from the **cls**, is limited. Within the terms of reference of the ELO project, only tools which have been designed for running on the ELO software- platform or tools which can be customized in the appropriate manner can use all services which are offered by the **cls** and the framework. However, it is intended and envisaged that, in the future, non-ELO tools would also be handled via the **cls** through some novel protocol / architecture.

A common attribute of non-ELO tools will be that they may appear to the ELO system as monoliths which have to be executed in form of a "black box". Their structure does not necessarily adhere to the "work-process" approach adopted by the project. These tools will have their own (supplied) user interface and dedicated help-desk. Nevertheless, the level of integration depends upon the particular features provided by those non-ELO tools. Non-ELO tools have to interface to the **cls** within an ELO-System. Such an interface can be achieved at three levels:

- Level 0
the non-ELO tool is a self-contained software-package which runs in a separate window, using its supplied user interface and help-desk.
- Level 1
the non-ELO tool can be started with a specific set of start-up parameters for instructing it to use specific input data or for triggering certain sub-functions which comply to the current dialogue context. As described for level 0 tools (above), the non-ELO tool will still use its own user interface and help-desk.
- Level 2
the non-ELO tool provides an internal programming or macro-language or a 'user exit' facility. These features can be used to restrict the active functions of the tool to the necessary subset for performing the currently active work process. Furthermore, the integration features of those tools may allow overloading of particular commands, such as the help command, with **cls**-like functionalities.

Each tool that is integrated into an ELO-System will increase the system's complexity (because of the additional functionality). That is, the provision of appropriate, integrated support for the ELO-Worker becomes more problematic. For this reason, it is recommended that only those non-ELO tools which comply to (integration) level 1 or 2 (above) should be integrated into an ELO-System. This enables their functionality to be restricted to the necessary, application-specific subset. If a level 0 type of interface were supported, it would increase the complexity of the ELO user-interface and thus violate a major principle adopted by the project. The decision to support only level 1 or level 2 types of interface preserves the transparency of the ELOS to the user.

12. The Computer Learning System (cls)

The **cls** is based on an architecture which make possible the provision of a flexible system for the ELO-Worker. In addition, the architecture provides for the integration of application and sophisticated support features, including help-desk and learning-sessions. The **cls** is designed as a set of self-contained tools which are described in the subsequent sections.

12.1 Objectives of the cls

The primary objective of the **cls** is to provide the ELO-Worker with an environment which allows them to acquire the necessary knowledge and skills for performing all supported ELO work processes. The support to be made available, including help-desk and learning-sessions, will be goal-oriented and is dedicated to the implemented "work-processes" defined in the requirements capture component of ELO.

The knowledge which is necessary to operate an application can be divided into three categories:

- professional (domain) knowledge,
- application knowledge and
- computer knowledge.

The **cls** provides the ELO-Worker with support in all three categories. But it should be noted that only "cognitive learning goals" will be supported (see (13)). As an example of the different kinds of cognitive learning goals consider the case of an insurance agent who has to check if a claim from a customer is covered by a the particular policy (the legal contract between the company and the customer). If the agent has questions concerning this type of policy, then he/she needs support aimed at filling this gap in his "professional knowledge". If the agent needs to know how to operate the ELO-System to carry out this check on claim coverage, then he needs to acquire "application knowledge". Finally, if the insurance agent has difficulties in locating a special function key on the machine then support is needed in the area of "computer knowledge".

12.2 Underlying Concepts of the cls

The **cls** is based upon three underlying concepts:

a) Results-oriented working style.

It is assumed that the potential ELO-Worker is in favour of (and willing to apply) a results-oriented working style. Therefore the **cls** and the application are structured according to the ELO-Worker's work processes. The architecture which has been designed for supporting this concept is described in the next section.

b) Flexible applications and user support.

The ELO-Worker will be provided with "views" on applications and support features, which are appropriate to their "level-of-expertise". The supported expertise-levels range from novice and beginner through intermediate to expert. The ELO-Worker's level-of-expertise is derived from an individual "student-model". (Note that an ELO-Worker's level-of-expertise may vary between different work processes.)

c) Integration of application and context sensitive help-desk plus learning-sessions. The **cls** provides the ELO-Worker with context-sensitive textual explanations and learning-sessions which relate to the current dialogue situation and assist the ELO-Worker in achieving the planned goal of the active work process.

12.3 User Interface and Contingent Support from the Architecture

The specification of the user's view of the system is derived both from the general requirements capture exercise and, in more detail, for the chosen ELO "trigger applications" of the "Presenter" and the "Macher(in)". The architecture of the ELO-System enables the ELO-Worker to view the system as a dedicated support tool for achieving a set of goals (with the methods to achieve each goal specified as a "work process") specific to the application domain. The underlying terminology can be summarized in the following points.

- A particular task or set of tasks, be these related to the ELO-Worker's professional domain, specific application or the computer system, can be viewed as "work-processes". The external representation of available and on-going work-processes are icons on the screen.
- The ELO-Worker initiates a work-process by selecting an icon representing the chosen work-process.
- The internal (**cls**) representation of work-process is a "frame". A frame is essentially a transition net describing the path(s) to achieve the work-process goal, the tools required and associated **cls** units.
- The functions associated with a work-process are executed by "tools" - these are a collection of executable objects. Tools fulfil application or **cls** relevant tasks. Tools are re-usable and can therefore be incorporated into different frames.

It is now appropriate to give a little more detail on the concept of a "frame", as this concept is central to the way a "work-process" is presented to the ELO-Worker. As stated above, the internal representation of work-processes are frames. Frames hold transition networks which describe the path for achieving the goal of a work-process and the tools required. Frames consists of "states" which are linked by "edges". The first state of a frame represents the frame-entry. The last state represents the goal of the associated work-process. The intermediate states represent sub-goals which are made up of certain data-constellations of particular sets of tools. States can be reached by traversing the edges between states. The process of traversing an edge - a transition - is implemented as a process of sending a message to the tool associated with a particular edge. Frames can "call" or make use of other frames. Due to the fact that tools can fulfil **cls**-relevant tasks and because of the concept of providing the ELO-Worker with work-process-oriented support in form of a help-desk and learning-sessions, frames can be used as the organizational measure for implementing both work-processes and learning-sessions.

A particular frame may (currently) exist in up to four different versions related to level-of-expertise: novice, beginner, intermediate and expert versions. No major problem is foreseen in extending the granularity of frames to finer distinctions in levels-of-expertise. The appropriate frame-version is determined according to the ELO-Workers's level-of-expertise. (This is based upon a technique used by Chin in the KNOVE system (2).

The ELO-Worker can initiate the help-desk or learning-sessions at any state of the execution of a frame. The **cls** offers the ELO-Worker various support-features:

- a help-desk with texts giving explanations of the current dialogue situation,
- an electronic reference manual,
- a description of the dialogue history,
- simulations and
- learning-sessions with different learning-methods.

12.4 cls Representation of Goal-Directed Dialogue / Work-Processes

Figure 5 shows a "frame hierarchy". It shows frames, enclosed in brackets, on three different levels. The states at a frame system start (S4, S6, S9, ... in this case) are called "frame entries", and the states at the end (S11, S12, S16, ... in this case) "frame terminators". In addition some frame entries give the option of changing the frame (expertise) level.

It is important to note that the same state can occur in different frames, as the same application themes can emerge from different application situations. The sum of all the frames represents the implicit domain knowledge. (The difference between implicit and explicit knowledge is in line with the definition suggested by L.A.Rendell (12).

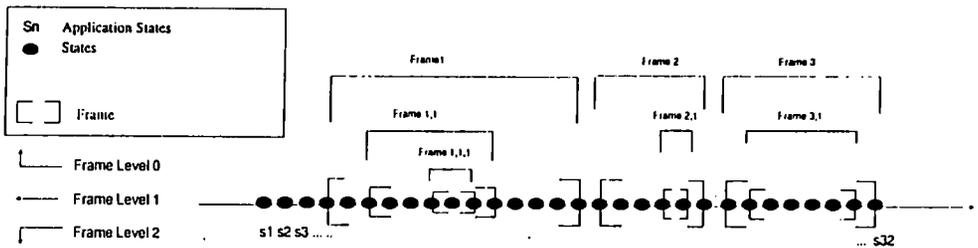


Figure 5: Frame Hierachy

The overall design of the **cls** is given by the **cls-Integration-Space (cIS)**, a three-dimensional space representing application (work-process sequence), qualification (flexibility / level-of-expertise) and user support (such as help-desk, glossary, learning-session, simulation). Each of these three aspects is represented by one axis of the cIS:

- the application axis(x - axis);
- the qualification mode axis(y - axis);
- the support axis(z - axis).

Moving through the cIS is equivalent to the ELO-Worker proceeding through the ELO-dialogue of the work-process. The cIS illustrates the integration between these three aspects of an application.

The application axis shows the sequence of states as they are arranged in frames. The execution of a frame, and ultimately the achievement of the associated goal, corresponds to the movement within the cIS along the application axis.

The qualification axis shows the flexibility of the system in relation to different levels-of-expertise. States and transitions between states are implemented for different levels-of-expertise so that different ELO-Workers are provided with frame-versions according to their individual level-of-expertise. Movement along the qualification axis is the responsibility of the "qualification-tuner", and the level is set at the start of execution of a frame. The qualification-tuner derives the ELO-Worker's level-of-expertise specific to the active frame from the "student-model".

The (user) support axis lists the available support-features from which the ELO-Worker can select the one that they feel is most appropriate in the current context. The ELO-Worker controls movement along the support axis via the user interface.

In cIS example given below, the application axis contains the states s_n of the frame sequence. The qualification mode axis has the levels-of-expertise "expert", "intermediate", "beginner", "novice". The support axis consists, in this example, of "textual help", "glossary" and "learning package-entries". The messages which enable transitions between states are shown as m_n . States of the help-text are denoted by h_n , of the glossary by g_n and of the learning-session by l_n .

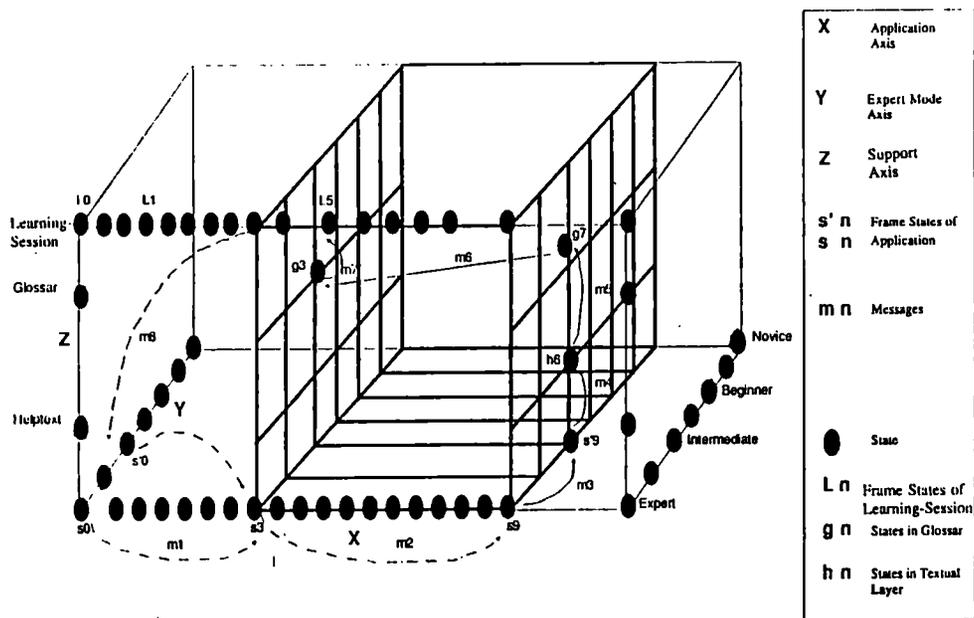


Figure 6: cIS Example

12.5 cls Tools

The **cls** consist of several components or tools which are, for the most part, invisible to the ELO- Worker. This section concentrates upon the visible components of the "help-desk", "student-model-tool", "qualification-tuner", and the logging and undo-facility. In addition, some information is given on the "Frame Manager", the "Frame Store", the "Logger" and the "Frame Interpreters". The help-desk is integral part of the user interface, whereas the other tools provide the user with an explicit view of their underlying (data-)models.

The help-desk provides the ELO-Worker with context-sensitive textual explanations which cover different aspects of the application. The ELO-Worker is offered:

- a detailed description of currently 'topical' error messages,
- a description of the elements shown on the screen,
- a description of the currently active work processes and the available options for proceeding with the dialogue and
- an explanation of the current user interface metaphor.

The techniques for presenting the help-texts include formatted texts (which may contain individually accessible elements) shown on the screen concurrently with glossary and hypertext-like features for maintaining references. The ELO-Worker can invoke the help-desk by clicking on a dedicated command-button.

Further support is given to the ELO-Worker by systematic and appropriate learning-sessions. The learning-methods which have been selected as being relevant to the implementation of these learning-sessions are:

- learning by instruction,
- learning by discovery,
- learning through reflection and
- learning by example.

(Further information on these chosen methods can be found in (1) and (8).

The student-model-tool and the qualification-tuner are jointly responsible for providing the ELO-Worker with a flexible system appropriate to the individual level-of-expertise. The student-model-tool collects statistics and other relevant information for modelling the ELO-Worker's behavior and their application relevant skills (14). The student-model differentiates between types of knowledge (domain, application and computer-knowledge) and the relevance of the collected information with respect to the time at which they have been acquired (frequency and recency). The student-model-tool maintains both long-term and short-term (session) storage.

The ELO-Worker's application knowledge is modelled separately for each frame. Frames are categorized into the three different difficulty levels: simple, mundane and complex (2). This is so that the complexity of work-processes is taken into consideration when interpreting the student-model.

The qualification-tuner derives the ELO-Worker's level-of-expertise (novice, beginner, intermediate or expert) from the student-model. This derivation is achieved by statistical means, and determines the next frame to be executed. In cases where no significant information on the ELO-Worker specific to the current context are available, the level-of-expertise is derived from a (user) stereotype. The ELO-Worker is provided with the means of viewing their own student-model. A dedicated tool allows the ELO-Worker to manipulate their qualification-level at any time.

The purpose of the Logging and Undo-Facility is to provide the user with a fault-tolerant application and to support the concept of (safe) learning by discovering, i.e. simulating a work process. The undo-facility will be used by the ELO-Worker as a conventional undo command. The difference from working with a simulation is that the undo-command refers only to the last entered command or a sequence of commands. The commands to be undone are not interpreted by the system in the light of a semantic context. For example, the command to send a fax might be 'undone', but this may have no impact on the environment if the fax has already left the ELO-System.

Simulations deal with work processes and therefore operate within a semantic context. Simulations are under user control, i.e. the ELO-Worker decides to 'simulate' the execution of a work-process and when to end the simulation. It is up to the ELO-Worker either to invoke the undo-facility in order to restore the dialogue-state (as it was prior to the start of the simulation) or to use the transfer the results of the simulation into the application (to use the results of the simulation as real application results).

Some parts of the **cls** are hidden from the user but it is important to give some brief outline of them for completeness. In response to a user action (or to a message from another frame) that another part of a work-process (or another work-process) is to be initiated, the Frame (Store) Manager selects, from the Frame Store the appropriate frame. A Frame Interpreter is then started to direct the work-process and associated interactions with that frame. Messages are sent to tools (objects) via a Message Router that provides both a check-point for security (user, tool and data access permissions) and an appropriate level for system logging operations. Logging operations, for undo and auditing, are recorded by the Logger.

13. Achievements to Date

Work undertaken and results achieved so far according to work area are as follows.

Applications Investigation and Requirements Analysis/Specification

including:

- investigation of distributed and mobile forms of work;
- identification of trigger applications for mobile IT&T (ELO trigger application = a complete information processing and communication system providing comprehensive support for the daily work of the target user groups);
- requirements capture, analysis and specification for the two ELO trigger groups and the ELO trigger applications encompassing requirements for activity planning and communication and support for workprocess execution.

Overall System Architecture Development and Selection / Specification of Generic Elements of an ELOS

including:

- specification of the overall architecture for an Elusive Office System (ELOS);
- specification and selection of generic elements and specification of the integration frame work for the 1992 ELO Demonstrator.

Specification and Implementation of Communication Components

including:

- specification and software and hardware implementation of an Early ELO COMMS Demonstrator offering an error-free data transmission over cellular radio networks to become part of the overall Demonstrator (currently additional functions are being implemented).

Specification and Implementation of Security Services

including:

- specification of data encryption/decryption, tool and data access, personal identification security services;
- implementation of the data encryption/decryption service which is intended to be integrated with the overall ELOS.

Design and Partial Implementation of Computer Learning System Components

including:

- specification, design and implementation of application-independent learning tools and generic functions and mechanisms for supporting the learning concept simulation, deduction, induction, learning by example and automated stepwise execution of a pre-defined application path

14. The ELO System (ELOS)

As a result of the activities of the first two project years we now have a clear view and specification of the ELO system, including:

- its detailed functionality,
- generic user interface specifications and @BULL1LICLO = concepts,
- software, hardware and communication components,
- security services to be integrated,
- the integrated computer based learning support.

Some of the functions of the software, hardware and communications are already implemented.

The technology strand of the project has identified the following technologies to support the aims of the ELO project:

14.1 Underlying Software:

- non-proprietary, open systems
- UNIXTM operating system, C + + , X
- ET + + (software toolkit)
- integration of non-ELOS programs (e.g.: MS-WORD)
- SQL-base relational databases such as Informix, Ingres, Oracle.
- Graphical User Interface (GUI)

14.2 Hardware / Communications:

- laptop including keyboard and pointing device
- cellphone
- cellphone voice, fax and data (German C-Netz for Demonstrator)
- access to fixed PTT services and LAN
- assessment of Pen User Interface devices
- voice input and storage
- scanner
- communications box integrating services providing high-level interface to laptop.

14.3 Project Software:

Object-Oriented Design

A Class Hierarchy is required to support the User Interface, Gopher (for a definition see below) and general Framework objects. Due to the limited project resources an existing class hierarchy has been taken to provide the basis for the functionality required.

Integration Framework

The Integration Framework will provide the mechanism by which disparate software components are integrated into the ELO environment. It will enable appropriate communication to take place between the local tools, resources and interaction facilities, and remote resources and tools, at the required level of security. The Framework will be sufficiently flexible to support the initial trigger applications, and will be smoothly extendable to support applications available in the future. Ideally the local and remote services should appear the same to the end user regardless of their location. It will be the interface between the learning system, the security services, the communications facilities, and the other ELOS's components. It ensures that tools are integrated as much as possible, taking into account the characteristics of their various interfaces.

The User Interface is an instance where the level of integration is obvious to the user. The choice of a particular development environment may well impact upon the user interface of the ELOS. In our case ET + + (15) is seen as the current front-runner which is strongly influenced by the MAC Applications style of working.

A functional representation of the Framework is depicted in Figure 7. It shows how the tools interface with the Framework and underlying ELOS facilities. Some of the possible message channels are shown. It is a conceptual rather than a functional representation so does not depict any control or data flow (11).

Gopher Family

The Gopher Family is effectively the "resource manager" for the ELOS. It knows the location of the various resources. It undertakes resource requests on behalf of local or remote tools. All requests are queued to be performed when the resource becomes available. Tools communicate with the Gopher Family by sending MTM messages from their Message Managers to the Gopher Family's Message Manager. This Message Manager allows communication between the Gopher Family and the computer learning system (**cls**) World. External requests are treated the same as internal ones, except that they arrive through a communications channel UFO (Underlying Function Object)

rather than the Message Manager. Each request is given a unique (across the whole organisation) identification tag containing originator, date stamp and id number (RAL91).

Security Services

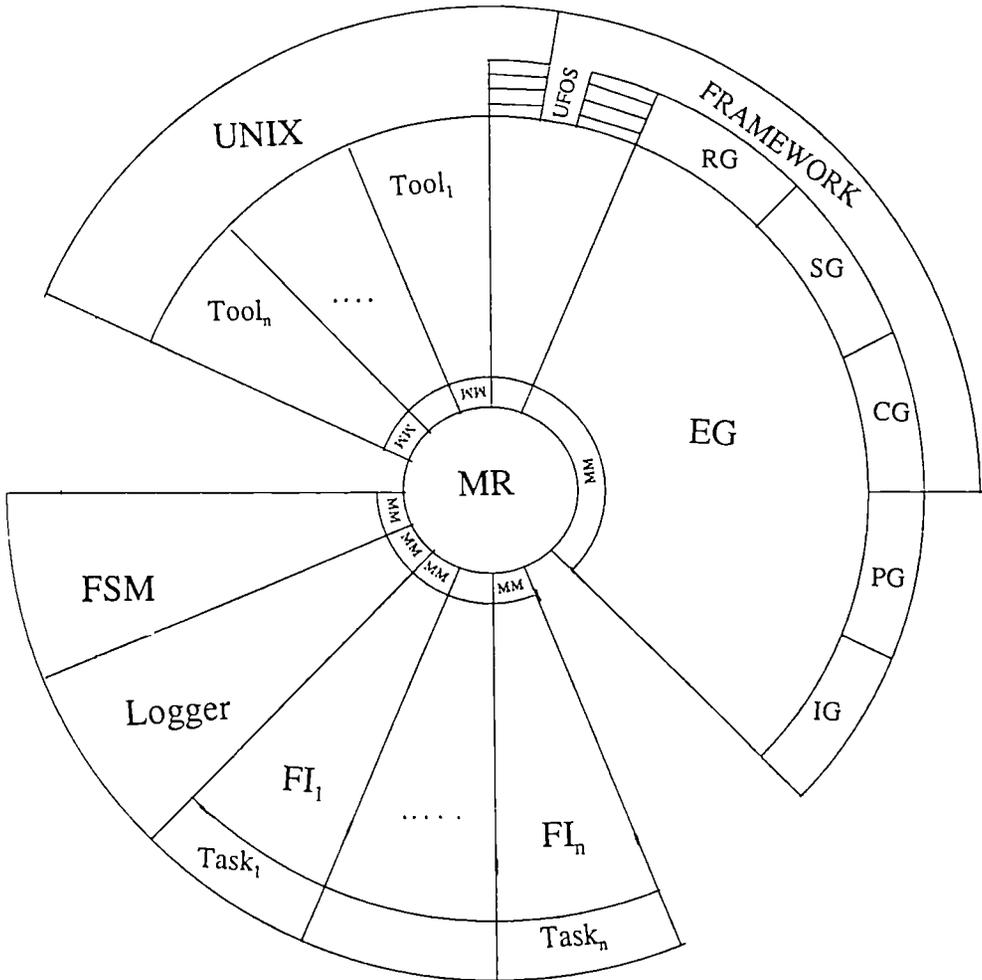
Within the ELO project the following security services have been specified:

- tool and data access control service
- end-to-end security service (data encryption/decryption)
- personal identification service.

The end-to-end security service, for which the strongest requirement exists among the ELO target user group (insurance agents) for which the ELO 1992 Demonstrator is being developed, has been implemented in a form compatible with the Integration Framework and communications hardware/software described above.

Computer Learning System

A highly integrated computer learning system that will support remote users is being developed within the ELO project. This novel feature of an ELOS will offer a range of assistance, from simple help to complex linked simulations of system objects under control ((3), and sections 11 and 12 above).



- UNIXTM underlying operating system and access to underlying resources (communications, disk-files, data-bases, ...)
- Framework the (Integration) Framework
- UFGOS Underlying Function Objects (supply access to/from underlying resources)
- *G the Gopher Family ("Resource Manager")
- MM Message Managers - handle communications/messages between the Gopher Family, object-oriented tools and cls components
- MR (cls) Message Router
- FSM (cls) Frame Store Manager (and Frame Store)
- Logger (cls) Logging system
- FI_n (cls) Frame Interpreter (supporting particular tasks/"work processes")
- Tools (object-oriented) tools

Figure 7: Functional Diagram of the ELOS (Integration) Framework

Specific Application Software

Specific application software - using the various software, hardware and communications components developed in the project - will be delivered. This application software will provide work process support to the target user group (insurance agents) for the ELO 1992 Demonstrator.

It was decided to select the work process of "claim handling" a frequent (i.e. daily) activity of the ELO Demonstrator Group. Detailed requirements for work process support for claim handling - forming the basis for software development - have already been analysed and specified by empirica.

Future

The following work is currently (May 1991) undertaken or will start in the near future:

- system design, (Integration) Framework development and implementation of the specified requirements in generic and specific applications,
- testing and evaluation of partial prototypes and the final ELO 1992 Demonstrator in mid- and late 1992,
- implementation of the remaining generic computer learning system (**cls**) components,
- specification and implementation of **cls** components specific to the 1992 ELO Demonstrator,
- specification and development of "supportware" to ease the use and introduction of ELO systems in different work environments,
- dissemination activities to raise awareness of the idea and concept of ELO and to investigate possibilities for further developing, marketing and exploiting the results of the ELO project.

The final ELO Demonstrator will be available in late 1992.

Acknowledgements

We wish to acknowledge the work of the ELO Consortium partners, especially the work of the ELO-Team at the Rutherford Appleton Laboratory, upon which we have drawn in preparing the chapter on "The ELO System (ELOS)" in this paper. However, nothing here should be taken to commit any partner, and the opinion and any errors are our own. We would also like to acknowledge the support of the Commission of the European Communities represented by Josephina Pijls and Attilio Stajano and all the other members of the "Advanced Business and Home Systems; Peripherals" Group and would like to thank the assessors of the drafts of this paper for their helpful comments and suggestions which we have incorporated into the current version.

References

- (1) L. Bolc, Computational Models of Learning, Springer Verlag, Heidelberg 1987
- (2) D. N. Chin, KNOME: Modeling What the User Knows in UI, in User Models in Dialog Systems, edited by A. Kobsa & W. Wahlster, Springer Verlag, Berlin 1989
- (3) Computer Lern Systeme GmbH: Generic Learnware Specification (B1.1/CLS/001.1) January 31, 1990.

- (4) empirica GmbH: ELOS Requirements Capture, Analysis and Specification for Presenters: Report I: Presenter Requirements of an ELOS (A.2/EMP/008.2) April 22, 1991.
- (5) empirica GmbH: ELOS Requirements Capture, Analysis and Specification for Presenters: Report II: The ELO Demonstrator Group (A.2/EMP/010.2) April 22, 1991.
- (6) empirica GmbH: ELOS Requirements Capture, Analysis and Specification for Presenters: Report III: The ELO Task Interface for Activity Planning and Communication (A.2/EMP/007.1) February 28, 1991.
- (7) empirica GmbH: ELOS Requirements Capture, Analysis and Specification for Presenters: Report IV: An ELO Demonstrator Group Work Process: Requirements for Claim Handling (A.2/EMP/009.2) April 22, 1991
- (8) H. Mandl & A. Lesgold, Learning Issues for Intelligent Tutoring Systems, Springer Verlag, New York 1988
- (9) Porat, Marc: The Information Economy: Definition and Measurement. OT Special Publication 77-12 (1), Washington 1977
- (10) SERC Rutherford Appleton Laboratory: Overall Architecture. Functional Split (Provisional Specification) (ELOPS) (A3.1/RAL) July 10, 1990.
- (11) SERC Rutherford Appleton Laboratory: Specific Issues. ELO Generic Elements (for the Demonstrator) (ELOGE) (A3.2/RAL) February 28, 1991.
- (12) L. A. Rendell, Conceptual Knowledge Acquisition in Search, in Computational Models of Learning, edited by Leonard Bolc, Springer Verlag, Heidelberg 1987
- (13) W. Steppi, CBT Computer Based Training, Klett, Stuttgart 1989
- (14) W. Wahlster & A. Kobsa, User Models in Dialog Systems, in User Models in Dialog Systems, edited by A. Kobsa & W. Wahlster, Springer Verlag, Berlin 1989
- (15) Weinand, A. /E. Gamma/ R. Marty: Design and Implementation of ET++, a Seamless Object-Oriented Application Framework. Structured Programming, Vol. 10, No.2, 1989

UNIX is a Trade Mark of AT&T in the USA and various other countries

RESOURCE MANAGEMENT IN THE HOME SYSTEM

DAVID G.J. FANSHAWE
Philips Consumer Electronics
Advanced Projects Group
New Road
Mitcham, UK

SUMMARY

To create a successful Home System market we must offer an expandable, and preferably self-configuring, system based on affordable products which will inter-communicate to offer cooperative features. This paper explains how we achieve the functional standardisation necessary to ensure interworking, whilst still supporting competitive market choice and allowing manufacturers to introduce new features.

INTRODUCTION

Consumer Electronics represent a multi-million ECU business for European companies. We are all consumers, and we have only to look around our own homes to see the impact of this rapidly growing industry. The television, the camcorder, the electronic keyboard, the microwave oven, and the cordless telephone are as commonplace now as the radio and gramophone were a generation ago. The microprocessor has revolutionised the industry, making it possible to add new features simply by changing the software. This allows manufacturers to react quickly to changing needs, and offer improved performance and simpler operation.

But sometimes we need several products to work together to provide the service, and each one has to be set correctly, and interconnected correctly. It is these "systems" - clusters of inter-connected products - which highlight the problems of user control. Arguably, the market is being held back by the consumer's concern that he will not be able to operate a more sophisticated product. A successful solution to system control will therefore lead to increased opportunities for European consumer electronics in the world market.

ESPRIT-HS: PROJECT 2431

The ESPRIT Home Systems project aimed to solve this problem of inter-operability. The objective of the workprogramme was to define how products shall be capable of interconnection and automatic intercommunication. This provides three major benefits to the consumer:

1 Remote operation

With all the household electronic equipment on a communications network, you can control anything from anywhere in the house; even from beyond the house via the public telephone system.

2 Automatic inter-working

Many of the things the consumer wants to do involve setting up several different products so they can work together. For example: copying a home movie; making

the room comfortable; cooking the dinner. If products can control each other, the consumer can push one button, sit back, and leave the equipment to sort it out.

3 Modular functionality

To some extent, present-day consumers can already buy products which give the one-touch features mentioned above. Double-deck videos, combination micro-waves and air-conditioners are examples of customised products aimed at specific needs. But while these have their place, the real need is to have modular products which can be automatically linked and configured to provide one specific service, and then automatically re-configured to serve a different need.

In professional areas, the concept of 'flexible automation' based on networked products sharing their resources is becoming quite common. So far this revolution has had little impact on the home, although even there the concept of separate products interlinked into a system is not totally unknown. Many consumers have an audio system comprising separate players, amplifiers, signal processors etc (usually interconnected by a complex arrangement of separate cables, requiring a significant amount of expert knowledge). Some consumers also have within their homes heating systems, security systems and telephone systems, generally requiring professional installation.

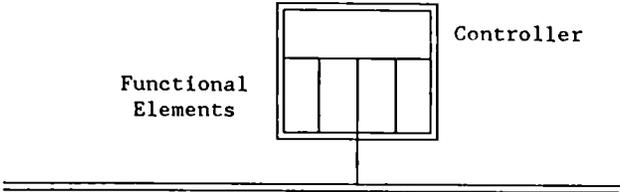
The Home System, as defined by Project 2431, represents a genuine advance for electronic products in the home. By linking the products together on a communications network, the householder gets a flexible and re-configurable system in which cost-effective modular products cooperate to provide useful and attractive services. Furthermore, with automatic set-up the consumer can add new 'plug-and-play' products without needing professional help.

RESOURCE MANAGEMENT

The Home System comprises the products and the network which links them. The products will not look very different from the familiar stand-alone products. They will have their specific functionality, and built-in features, just like present-day products; but through the network their 'application resources' become available to all other products connected to the network.

Each product contains a communications chip which allows it to communicate with other products in the home; but getting products from different manufacturers to communicate is only the start. The message must also be understood (which implies standardisation at the functional level), and the application resources must be shared fairly between competing processes. This aspect of system management is a special feature of the ESPRIT-HS architecture.

With consumer products becoming ever more complex and multi-purpose, we need a model capable of representing not only all present-day products, but (hopefully) future ones as well. In the ESPRIT-HS architecture we decompose real products into 'devices' and 'controllers'. The devices are the functional elements of the product, while the controllers provide the features.



Each device offers one 'service', or 'application resource'. Most low-cost products offer just one service (the light bulb simply provides illumination), while many expensive products offer several services (the TV may offer pictures, sound and teletext). But some expensive 'high-tech' products may offer just one service (e.g. the CD player), while some low-cost products offer several services. Let us take the telephone as an example. It can be modelled as a set of five application resources:

- 1- Sound detector (the mouthpiece)
- 2- Audio output (the earpiece)
- 3- Audible alarm (the bell)
- 4- User interface (the keypad)
- 5- PSTN gateway (phone company connection)

These are connected to a microprocessor - the local controller, which provides the basic telephone functionality. It may also offer additional features like 'last number re-dial', '1-touch emergency calls' etc. By connecting this product to the home network, all its application resources become available as general-purpose components in the home system. The telephone bell can be used as a 'wake-up' alarm; the sound detector can be used as a baby cry detector; the earpiece might provide quiet background music; the keypad may control the TV set; and the PSTN gateway may allow the owner to control his household products from any public telephone. The parts do not have to be made as separate units, the only requirement is that the modules can be separately configured. It is this configuration modularity which opens up exciting prospects for users and manufacturers alike.

In order to exploit this market opportunity we need a system architecture which allows efficient sharing of application resources. Products are designed in a modular way so that each application resource is provided by a single "Device". The 'features' in all modern products are provided by a microprocessor. This is the "Controller" element in the product. The advantage of a modular architecture is that controllers can control devices anywhere in the system. This modular concept of Devices and Controllers lies at the heart of ESPRIT-HS architecture.

SIMPLE DEVICES

The "Simple Device" (SiD) represents the simplest entity (least software, and therefore lowest cost) in the ESPRIT-HS architecture model. Simple Devices respond to functional commands which control their application resources. A SiD light will respond to the command from a SiD switch to turn on. Standardised commands will make a SiD telephone bell ring, and a SiD tape deck start to play. Some SiD's may respond to a wide range of commands, but they are still classified as "simple" because they just do as they are told. They are not 'aware' of running processes. They do not have any 'system management' capability.

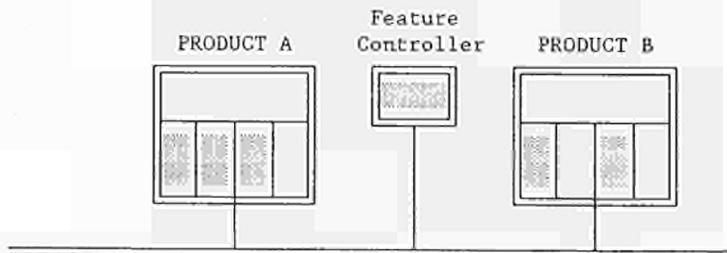
FEATURE CONTROLLERS

Successful products sell because of their features. Automatic features (e.g. a sequence of commands in response to certain input conditions) are usually provided by a microprocessor which is managing the system resources, and thus automating the process which the consumer would have to supervise in a non-automatic product. In the ESPRIT-HS architecture model we define the "Feature Controller" (FC) as that entity which is responsible for controlling the application resources in order to provide the feature. For example, "Record TV" is a feature whereby when the consumer presses

the RECORD button on the video recorder, it automatically starts to record whatever is showing on the TV screen. This needs coordination between two products, the TV (which knows what channel it is tuned to), and the video recorder (which needs to tune to that same channel). Many products will include a feature controller, but since FC's can control devices anywhere in the system, the feature controller in one product can use the application resources in another product.



A feature controller may become a product in its own right - adding extra features to an existing product, or coordinating the application resources of several products in order to offer new services for the consumer. In practice, feature controllers could be quite small and cheap to produce (they only need a microprocessor, memory and network interface).



The consumer can add extra Feature Controllers (to add extra features). In most cases the FC output will control devices, but in some instances the output from one FC may feed into another FC. One example is upgrading a security system. The consumer may initially buy a low-cost system comprising some detectors, some audible alarms and a feature controller (FC1). When the consumer wishes to upgrade (e.g. reducing false alarms by applying more sophisticated signal processing on the detector outputs), this can be done by simply adding an extra feature controller (FC2) which intercepts the detector output signals, stops FC1 acting on the raw data, and processes the signals. If it decides the event is genuine, it will send a signal to FC1 to activate the original alarms.

The feature controller thus represents an important area for new product opportunities. It is a means of allowing the consumer to upgrade the functionality of the system without having to throw any equipment away. In some cases the new functions may be specialist features, designed to appeal to a sector of the market (the elderly, the disabled, the hobbyist etc). Such niche markets can be handled economically in a feature controller.

COMPLEX DEVICES

There may be situations in which different feature controllers are sending conflicting commands to the same device. For example, the video recorder might be used to record visitors who come to the house when it is unoccupied. But this should not be allowed to interrupt a timed recording: UNLESS it is a real emergency (e.g. an intruder is breaking in), in which case it should take priority. Here we are running into real-time conflicts, and the device must be capable of managing its application resource in response to priorities. Simple Devices (by definition) cannot resolve this conflict. To do so needs a more intelligent entity, the Complex Device (CoD). This unit has the extra capability of managing its own resources, e.g. allowing one FC exclusive use of its services until the process has finished. This ensures that processes do not get interrupted by less important requests.

DEVICE COORDINATOR

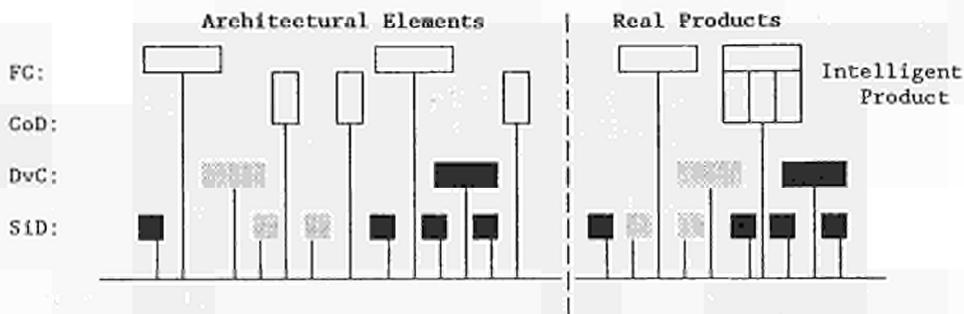
The Device Coordinator (DvC) provides resource management for a group of Simple Devices. This provides an up-grade path, because a Simple Device plus a Device Coordinator is equivalent to a Complex Device. It is the most cost-effective way of providing resource management in a system which has a lot of simple products. For example, we may decide to automate the domestic lighting system, not only to have convenient remote control of light level, but also to include automatic sensors which save energy by switching off unnecessary lights (e.g. when the daylight is adequate, or when the room is unoccupied). We may also add a security product which simulates occupancy by randomly switching lights on and off. There are a lot of lights in the house, so we don't want the expense of a Complex Device in each one. However with simple devices there will be a conflict between the Occupancy Simulator which is trying to turn the light on, and the Energy Conserver which is trying to turn it off. The DvC provides this management function, allowing the lights themselves to be low-cost Simple Devices.

THE MODULAR SYSTEM

We have defined three basic entities: the Simple Device (SiD); the Device Coordinator (DvC) and the Feature Controller (FC). All real products can be modelled in terms of these three elements. In practice, we find that most real products are designed in a modular way, and each functional module is a SiD plus a DvC, so we define this as a fourth entity: the Complex Device (CoD). We can therefore model the 'Intelligent Product' as a set of Complex Devices plus a Feature Controller. This model therefore fits in with present-day products and design practice; but it also allows us to model (and thus optimise) the functional elements within the products.

ENROLMENT

In an intelligent system it must be possible for a Feature Controller to discover what application resources are available on the network. In a practical home system, new products may be added - or removed - at any time. Whenever a feature controller wants to check what application resources are present it broadcasts an ENROLMENT_START message, and all the devices (in turn) respond with their DEVICE_DESCRIPTOR which infers functionality. The device will also give its UNIQUE_CODE, if the manufacturer has



implemented this optional feature. This allows a device to be recognised even if it has been moved to another part of the network (e.g. a portable computer or a telephone).

CONTRACTING

Enrolment allows flexible automatic system configuration, but at the cost of some software complexity in the product. For "intelligent" products this represents a marginal cost, but for "simple" products it might impose a severe cost penalty. Simple devices may therefore CONTRACT this responsibility to a Device Coordinator. The device coordinator then negotiates with the Feature Controllers, and also controls the availability of the SiD's resources through a token mechanism (see below).

SEMAPHORE

It is neither practical nor future-proof to envisage a single central controller managing every activity in the home system. The Feature Controller concept provides a fully distributed control mechanism, which fits in with the "buy only what you need" philosophy which is at the heart of all successful consumer products. But this does mean you may have two controllers competing for the same resource. Many services require the resource for a period of time, and we need a mechanism to ensure that running processes do not get interrupted. This is done by a semaphore mechanism.

Each application resource has a logical token. The Feature Controller can only control that device if it holds its token. By continuing to hold the token, the Feature Controller prevents any other feature controller interrupting the process. Feature Controllers contend for tokens through a PRIORITY mechanism. Eight operational priority levels are supported, from IDLE (lowest) to SAFETY (highest). There is an even higher level (INSTALLER) which is only used during installation and maintenance.

CONCLUSION

Under ESPRIT-HS architecture we model the product as a set of Functional Elements (which provide the application resources) coordinated by a Controller. There is a finite set of functional elements (SiDs, CoDs, & DvCs), and these provide the basic functionality of the products. The features (which is where brand identity and competitive advantage come in) are built into the feature controllers. In this way we achieve the functional standardisation necessary to ensure interworking, whilst still supporting competitive market choice by allowing manufacturers to introduce new features.

THE ARGOSI PROJECT FOR ISO/IEC GRAPHICS AND NETWORKING STANDARDS

J.-J. Bardy†, R.A. Day‡, D.A. Duce‡, J.R. Gallop‡, L. Mistralt, D.C. Sutcliffe‡

† Thomson CSF
160 Boulevard de Valmy
F-92704 Colombes

‡ Informatics Department
Rutherford Appleton Laboratory, UK

ABSTRACT

This paper summarizes the major technical achievements of the ARGOSI project to June 1991. The objectives of the project are to advance the state of the art in the transfer of graphical information across international networks and to improve the quality and applicability of standards in this area. The project has devised a classification scheme for applications in terms of their requirements for graphics and networking services. As a result of this a prototype application was selected for an application demonstration which will show how the CGM and FTAM standards can be used together to enable remote access to individual pictures within a CGM. This uses a CGM FTAM document type developed within the project, which is now being standardized. The project has also contributed to an EWOS Technical Guide defining a mapping of the X Window System onto an OSI stack. An ASN-1 description of the SGM standard's functional description is also being defined within the project.

1. INTRODUCTION

Esprit (European Strategic Programme for Research into Information Technology) Project 2463, ARGOSI - Applications Related Graphics and OSI Standards Integration, started on 1 March 1989 and is of 3 years duration. Eleven organizations are now involved in the project:

Prime Contractor	Thomson-CSF	France
Partners	INRIA	France
	RAL	UK
	Hitec	Greece
	Tecsiel	Italy
	COSI	Italy
	GMD-FOKUS	Germany
Associated Partners	Laser-Scan	UK
	FhG-AGD	Germany
	GESI	Italy
	University of East Anglia	UK

The project arose from the recognition that within ISO/IEC, as elsewhere, standards for computer graphics and standards for Open Systems Interconnection have been largely developed in isolation and little serious attention has been paid to issues of integration. The objectives of the project are twofold:

- Advance the state of the art in the transfer of graphical information across international networks.
- Improve quality and applicability of standards in this area.

The project is expected to make an impact under a number of headings:

- (1) *Knowledge*: when the project started, there was little understanding of the requirements which graphics place on networking in terms of the particular networking services required to transfer graphical information across networks. Such requirements arise, for example, because of the structure inherent in graphical information and the need to access subunits within an overall structure.
- (2) *Standardization*: international standardization activities are seen as a key mechanism for achieving coherence and interoperability in an otherwise fragmented area. The contribution of the ARGOSI project to standardization activities will be to identify requirements for future graphics and networking standards and make appropriate inputs to the standardization committees concerned. The organizations involved in the project have a long tradition of contributing independently to graphics and networking standardization. ARGOSI provides the opportunity for these organizations to work together in the standards-making process and so to form the nucleus around which a more coordinated and coherent European input can be made to the international process.
- (3) *Industry*: from an industry standpoint the project sets out to demonstrate what can be achieved in distributed graphics using existing international public data networks. This will point the way to what can be expected as better international networks become available in the future and to the requirements for such facilities from applications. To demonstrate the state of the art, an application demonstrator is being constructed. This is described later in this paper.
- (4) *Commercial*: the project will develop software tools for graphics and networking services which will form the basis directly, or indirectly, of commercial products. The prototype demonstrator will also generate knowledge and experience of how to (and equally importantly, how not to) build distributed graphics applications using ISO/IEC standards.

The remainder of the paper describes what has been achieved so far in the project, concentrating on the prototype demonstrator and related activities.

2. METHODOLOGY

As the project title implies, the project is driven by applications and their requirements in terms of graphics and networking services. Effort in the project has been devoted to classifying applications according to these requirements. This work is briefly described in the section *Classification of Applications*.

A Study of Services task is looking at how the requirements emerging from the Classification activities can be satisfied by existing ISO/IEC graphics and networking standards and to what extent new standards or amendments to existing standards are necessary. The results of this activity will be fed into the appropriate Standards bodies and committees as they arise. The work to date is described in the section *Study of Services*.

To demonstrate what is possible now with existing or emerging graphics and networking standards and international networking facilities, an application demonstra-

tor is being developed. Before the demonstrator could be defined, it was necessary to ascertain what bandwidth could be delivered between partners' sites by public data networks. This work is described in the section *Connectivity via Public Data Networks*. The demonstrator itself is described in the section *The Application Demonstrator*.

Finally, the project will assess the success of integration achieved and will identify those areas where future standardization activities are required.

3. CLASSIFICATION OF APPLICATIONS

The principal aim of this part of the project was to investigate requirements for graphics and networking services in combination and to present this information as a taxonomy of applications in terms of these requirements. To establish such a classification scheme it is essential to derive an understanding of how current applications use graphics and networking services in combination, together with an understanding of how requirements will evolve in the future. Requirements can never be totally divorced from technology and so it is also necessary to have an understanding of how the underlying technology will evolve, leading to speculation on how existing applications can use improved technology and what new applications become possible because of improved performance and other characteristics.

The method of data collection adopted has been:

- (1) *Interviews*: visits to organizations developing applications which use graphics and networking, either now or in the future. Technical people in these organizations have been interviewed. A questionnaire was constructed to guide the interviews.
- (2) *Postal survey*: to gather information from a broader spectrum, a postal questionnaire was constructed and sent to a wide range of organizations, world-wide.

Interviews covered a range of application areas. Organizations in the countries represented in the project were interviewed.

Meteorology will be taken as an example. Several organizations working in this area were interviewed. Weather forecasting is inherently distributed. Raw data is received by forecasting centres from remote sites and forecasts are disseminated to a wide range of geographically distributed sites, both national and international. The data disseminated consists in the general case of satellite images or background maps with graphics overlays (eg contour maps) to present the forecast. The Computer Graphic Metafile (CGM) - a metafile for the storage and transfer of picture description information standardized in ISO/IEC 8632) - is starting to be used for representing such graphical data. One of the main problems in this application area is that data volumes are very large (for example satellite images and complex contours) and since the graphics standards do not in general make statements about the data volumes an implementation must support, many implementations are not able to handle the data volumes involved.

At the present time, much of the dissemination of forecast data is done by fax machines, but the advent of public data networks is heralding a move towards dissemination by transfer of CGMs of parts of CGMs. A CGM represents a sequence of pictures and there is a requirement from this application area to be able to transfer individual pictures in a CGM as well as a complete CGM.

By use of the interview and postal data a preliminary taxonomy for the classification of distributed applications according to their graphics requirements has been constructed.

The key idea in the taxonomy is to look at the points in a distributed application at which graphical data are transferred between co-operating components, and then to classify these according to the graphical information being transferred and the network services required to effect the transfer. The points at which data are transferred have been called *associations*. An association is defined as a connection between two components for the purpose of conducting a series of "transactions". The association may or may not be closed down at the end of this series of transactions, depending on the circumstances in effect. For example, the association may be kept permanently open, in some circumstances, or there might be a defined end-point to a series of transactions, after which the association is closed. Alternatively, there may be a more complex strategy involved in setting up and closing down of associations.

This concept of an association has a similarity to the concept of an "application association" described in the OSI Reference Model (ISO 7498), and in more detail in the OSI Association Control Service (ISO 8649). However, the two concepts should NOT be equated - as used here an association is intended to describe the most general type of network connection, be it a full OSI application association or one based on other networking models and protocols. At present, however, the taxonomy does regard an association as having a defined beginning and end (which would map onto association "establishment" and "releases" in OSI terminology). The possibility of atomic (i.e. "single shop") network transactions, with no concept of association establishment is not catered for. This reflects the nature of the applications studied when deriving the taxonomy.

The taxonomy is constructed from a set of *metrics*, each of which has a small number of values. In this way, with n metrics, an n -dimensional matrix can be constructed, and an individual application's association can be placed in a single 'cell' within the matrix, depending on the values of the metrics taken by the association. It should also be possible to assign appropriate combinations of graphics and OSI standards to each cell, and hence to advise on which standards are applicable to the part of the application categorized into that cell. If a set of standards cannot be associated with a cell, this is an indication that new standards may be needed.

The metrics currently defined are as follows.

- *semantic content* - a description of the graphics data flowing across the association, and categorized in terms of seven levels of abstraction (APPLICATION, GRAPHICS-RELATED, CONSTRUCTION, VIRTUAL, VIEWING, LOGICAL and PHYSICAL). The last five are taken from the ISO/IEC Reference Model for Computer Graphics (currently at Draft International Standard stage);
- *dimensionality* - a categorisation of the graphics data that can refer to both Euclidean or non-Euclidean geometry, and may also include time represented as an extra dimension;
- *data source* - a description of whether the data to be accessed are a complete, previously-defined body of structured information (such as a complete file of stored information), or whether the information flowing across the association is being used to construct or interact with a picture;
- *control* - a description of which end of the association has control over the selection of data and initiation of transactions conducted over the association. It also characterizes dynamically from other data;
- *selection criteria* - a description of the parameters that might affect the choice of networking facilities. Currently data volume, transaction time, transaction interval and acceptable error rate have been considered.

To give a flavour of the classification scheme, Table 1 shows the classification of several different applications, leading to different boxes being used in the taxonomy framework. There is not space here to explore the meanings of the values of the metrics in more detail.

Metrics	Applications					
	UK met. pred. region (1)	Applic. prototype (2)	UK met. not pred. region (3)	Thomson Ocean Traffic (4)	RAL remote pic. lib. (5) Overview One pic.	
Semantic Content	VIRTUAL	VIRTUAL	CONSTRUCTION	PHYSICAL or VIRTUAL	VIRTUAL	
Dimensionality	2D	2D	2D	2D	2D	2D
Data Source Control	COMPLETE GIVE-ME	COMPLETE GIVE-ME	COMPLETE GIVE-ME GIVE-ME	PARTIAL TAKE-THIS or	COMPLETE GIVE-ME	COMPLETE GIVE-ME
Selection Criteria	WHOLE	STATIC-SUBSET	DYNAMIC-SUBSET	WHOLE	STATIC-SUBSET	STATIC-SUBSET
Quality of Service: Data (6)	(4, 1)	(3, 2)	(4, 1)	(2+, 0)	(6, 0)	(4, 2)
Volume	MINUTES	SECONDS	MINUTES	MINUTES	SECONDS	SECONDS
Transaction Time	DAYS	MINUTES	DAYS	MINUTES	MINUTES	SECONDS
Transaction Interval	NO-ERRORS	NO-ERRORS	NO-ERRORS	NO-ERRORS	DONT-BOTHER	NO-ERRORS
Error Rate						
Comments	large volume	complete, static background	large volume	static background		
Application area	Meteorology / Cartography	Traffic information	Meteorology Cartography	Traffic control/ information	Education	Education

Table 1 - Classification of Applications

Notes

- (1) Application 1 is from the UK Meteorological Office. The meteorologist request a contour map for inspection at a local workstation. The regions of interest are predefined.
- (2) Application 2 is the ARGOSI demonstrator. There is a static background and other local graphics needs to be merged with the transmitted picture.
- (3) Application 3 is like application 1, except that the region requested need not be a predefined one.
- (4) Application 4 is an ocean traffic control application arising from an interview conducted by Thomson. A ship can display the tracks of all ships in the region under control. There are a number of design alternatives which can lead to different values of the metrics. There is a static background.
- (5) Application 5 arises from the coordination of graphics in the UK academic community. One specific problem is the provision of a network-accessible picture library. It should be possible to request a few specific pictures. It should also be possible to request that all pictures in a library be displayed as one image. For this complete image there is a design issue concerned with whether it is preferable to construct the image remotely and only transfer the remote image.

A number of applications are now presented which fall into the same box in the taxonomy framework. This box is denoted by:

Semantic content VIRTUAL
 Dimensionality 2D
 Data source COMPLETE

Control	GIVE-ME
Selection criteria	STATIC-SUBSET

In a number of cases, the application was still at the design stage at the time of the interviews. In these cases, the application could be placed in different boxes depending upon the design decisions; these alternative boxes are identified together with the design alternatives that led to their use.

Distance Learning

In a proposed distance learning project (Italy), the box denoted the set of values of the metrics corresponding to a situation where a student initiates the transfer of graphical information to his workstation. If instead, part of the distance learning application is executed at the student's workstation, the semantic content is APPLICATION-DATA. If the course-ware centre initiates the transfer, the control metric is TAKE-THIS.

Manufacturing

In a manufacturing application under design in the UK, the operator carries out a wiring task, guided by a picture transmitted from a central location. The operator requests a picture when ready for the next task. The selection criteria could be regarded as WHOLE - the whole of the graphics information for the task is delivered; or the term WHOLE might correspond to a set of operating tasks, in which case the picture for a single task would correspond to STATIC-SUBSET.

Power Station CAD

This application in France involves the enquiry of a large CAD database over a wide (national) area. The user initiates the inquiry of the database. The semantic level depends on the degree of manipulation required in the user workstation. It is anticipated that the next version will use CGM. If the database consists of graphical information, it is likely that the data source metric would be COMPLETE. If on the other hand, the database contains application-oriented information but the network management transmission takes place at the VIRTUAL semantic level, some central processing takes place and the data source would be PARTIAL.

The full details of the classification scheme are contained in^{2,3}. The work has led to an improvement in the understanding of application needs in a number of areas.

- (1) The taxonomy provides a methodology for analysing application requirements for distributed graphics.
- (2) Recent work in the task has confirmed the importance of understanding a distributed application, by creating multiple associations where necessary.
- (3) The taxonomy has been used as a design aid, to analyse possible scenarios.
- (4) In relation to standards, the importance of subsets of information was revealed in a significant number of applications. The appearance of this early on in the task gave added weight to the Project's work on CGM over FTAM and subsequent work has confirmed the importance of subjects selected before transmission. This question is in principle not restricted to CGM. Subsets are, for example, relevant to the standard for imaging, being proposed in ISO/IEC JTC1/SC24.

After a certain amount of elaboration of the taxonomy framework, confidence is being gained in the set of metrics that has been derived. In some cases, however, the set of values for a metric may need further refinement. For example, it may be necessary to distinguish between three Cartesian coordinates plus time, and four dimensional homogeneous coordinates within the value 4D of the dimensionality metric.

The quality of service metric needs further study, for example, how does it relate to the taxonomy framework. Consideration needs to be given as to whether image fidelity should be included as a component and whether this covers the requirements for image compression. There is also an issues of whether geographical area is an important aspect of quality of service.

The control metric also needs further study. It is important to understand that "control" here is used in the sense of the style of control required by applications. A particular style of control at this level might be realized by networking services which exhibit other styles of control at lower levels in the network architecture. There is a scope for a deeper understanding and characterization of association control.

The motivation for constructing the taxonomy was to classify the requirements of applications so that they could be studied to see if they were met by the current set of Graphics and OSI Standards. However, while the work was being carried out, it was discovered that many applications could be implemented in a number of different ways which used different graphics and networking services. The taxonomy framework can be used as a design aid, by exploring how different architectures map onto the framework. The framework and its mapping onto required services can then be used to examine the trade-offs between different architectural solutions. There is scope for further work here in exploring the effectiveness of this approach, and for putting the trade-off analysis on a more quantitative footing.

The work has also brought to light a number of areas which no combination of existing standards address, or are only partially addressed. An example of the latter is the support of interactive working using the CGM standard for which no harmonized networking services support currently exists. An example of the former is a graphics standard for the transfer of a dynamic subset of data. There is a link here to work in spatial data management which is worthy of further exploration.

The classification work has made three definitive inputs to the remainder of the project. The first was to identify an application for the prototype demonstrator. The application chosen is an information system for European transportation road freight operators (lorry drivers!). The application is representative of a number of the application areas considered and will use CGM and FTAM (File Transfer Access and Management - ISO/IEC 8571) as the basis for the graphics and networking services required. This is described further in section 5.

The work has also identified the need for a standard way in which to map CGM onto FTAM (see section 6.2). Finally the work has shown up a strong interest in using X Windows over OSI networks and work has been initiated in collaboration with other interest groups to define a mapping of X Windows onto OSI.

4. CONNECTIVITY VIA PUBLIC DATA NETWORKS

The wish to demonstrate the prototype application in as general a context as possible, means that it should be possible to demonstrate it across international boundaries (to demonstrate the 'European' nature of the work) and between as many partner sites as possible (to demonstrate the 'Open Systems' nature of the work). These aims dictate the use of publicly available facilities - the usage of private, leased lines (for

example) would be unfeasible within the resources of the project. It is also an aim of the demonstrator to show that publicly available facilities *can* be used to offer useful services in conjunction with ISO/IEC graphics and networking standards. One result of the demonstration work should be to identify the weakness of currently available facilities and to make recommendations as how these can be overcome. To do this 'real' service conditions must be experienced.

The connectivity work established what X.25 facilities could reasonably be made available at each site and carried out a series of experiments to estimate the bandwidth available between sites and estimate the reliability of connections. Attention was also paid to previous connectivity studies, most notably the COSINE studies of performance and reliability of Packet Switched Public Data Networks (PSPDNs) (COSINE Study 6..2.2).

It was decided that the quantities to be measured as a means of gauging suitability of the connection should be kept simple, viz:

- throughput obtained, both the user and total (at the network level);
- call setup times for services;
- success rate for initiated test sessions;
- detailed statistics for services that failed for services that failed (reason for call clearance etc).

A number of test methodologies were considered, based on which application-level software was to be used to conduct the tests. It was decided that the limited effort available dictated the use of the only software common and available to all sites - X.29. It was also decided that testing would be done by listing a large file (approximately 150 Kbyte) over a terminal session set up via X.29. This would provide reasonable throughput and call failure statistics, but would not, however, yield call setup time statistics, due to the interactive nature of the sessions involved.

The PSPDNs involved in the study were as follows:

DATEX-P	the PSPDN operated by the German PTT;
HELLASPAC	the PSPDN operated by the Greek PTT;
ITAPAC	the PSPDN operated by the Italian PTT;
JANET	a packet switched network set up as a common resource for the academic community in the UK, but not for use outside this community at present;
PSS	the PSPDN operated by one of the UK PTTs (British Telecom);
TRANSPAC	the PSPDN operated by the French PTT.

At the time the tests were carried out, Hitec were still awaiting a connection to HELLASPAC. This is now in place. RAL used a connection to PSS routed through JANET for the experiments, but a direct connection has subsequently been installed.

The results of the experiments show that a general estimate of 3Kbits/s is reasonable for the throughput to be expected between the Partner sites. This is in broad agreement with the COSINE file transfer study - the figures are slightly worse, possibly due to the inclusion of ITAPAC sites in the present study. The best throughput figures were between TRANSPAC and DATEX-P. Figures to and from PSS were acceptable. Reliability results were encouraging. Network congestion was the most frequent problem, most noticeably at INRIA for incoming calls and at Tecsiel for outgoing calls. There did

seem to be evidence of problems with international connections to ITAPAC, with Teciell (Pisa) faring better than COSI (Milan) overall.

5. THE APPLICATION DEMONSTRATOR

The application demonstrator will take the form of a Geographical Information System. The aim is to model a system whereby land-based freight operators may obtain information on likely transport difficulties due to adverse weather, roadworks etc in any European country. This information will be distributed via wide-area networks, with the assumption that it is effectively publicly available. At the time of writing the design of the demonstrator is complete and implementation is in progress.

The architecture of the main unit of the application, the consultation unit, at the display point (a national site) is shown in Figure 1. The basic model is one whereby static information (road maps) will be stored at the display point. There will be a database of 'difficulties' associated with each country, held at one site in each country. The user software will interrogate the database to obtain the relevant information and overlay this graphically on the appropriate map. Thus a relatively small amount of information will need to be transferred in a transaction. By using the correct remote service, no application code is needed to serve the difficulty data.

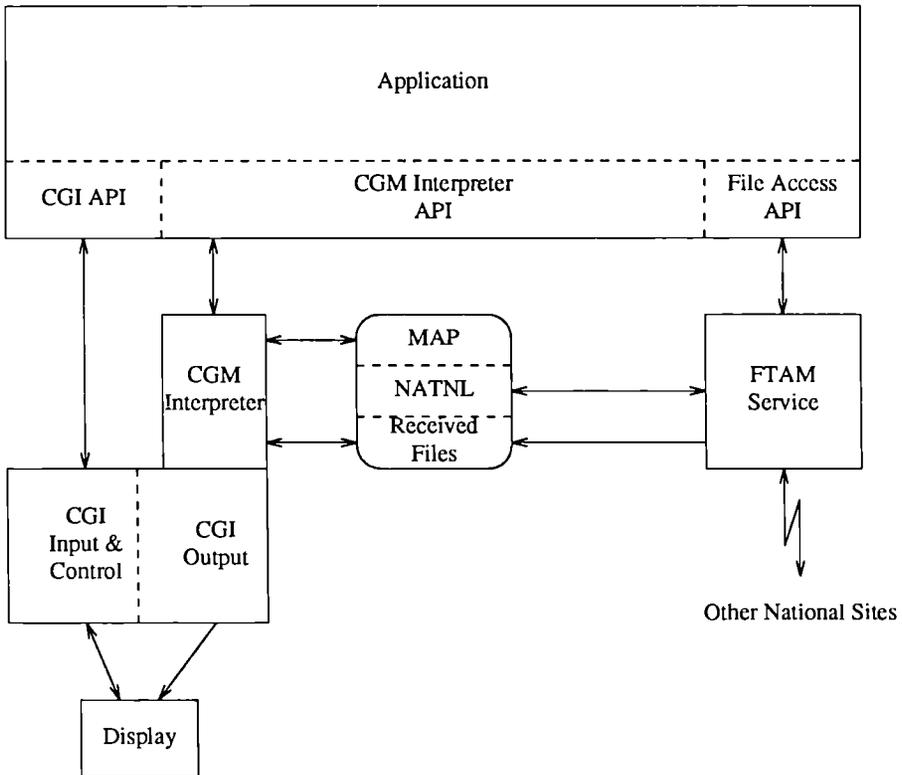


Figure 1: Architecture of the application consultation unit

The data will be transferred as a CGM, or more precisely, they will be abstracted from a CGM held in the 'difficulties' database. To do this, the CGM structure is mapped onto an FTAM representation, and FTAM is used for the data transfer. It has been shown

that the desired performance requirements for queries can be satisfied with the bandwidth of 3Kbit/s which PSPDNs can currently provide between partner sites.

The graphics architecture consists of a CGM interpreter and a subset of the CGI (the Computer Graphics Interface - ISO/IEC Standard 9639), to display CGM pictures, in combination with CGI input facilities.

A programming interface to FTAM is being developed, based on the MAP 3.0 API. This is a definition of an API that has been placed in the public domain by MAP (the Manufacturing Automation Protocol initiative - an industry-led consortium dedicated to the definition of suitable functional standards abstracted from ISO/IEC OSI, Graphics and other standards for the purpose of achieving a closer integration of the manufacturing process). At present ISO/IEC work in this area has not advanced to a stage where it can be used.

The application demonstrator will use an API based on the *context free file copy service* subset of the MAP API. This provides a simple means of transferring FADUs (File Access Data Units, structured elements of a file). It has been found necessary to extend the service offered by this subset to allow asynchronous operation ie the ability to have underway simultaneously several file transfers, all initiated from a single process via the API. (The MAP API allows only one such initiation at once, the initiating process then being blocked until the transfer is complete). This has resulted in the addition of an extra service to the *context free file copy service*, to allow the status of transfers in progress to be checked. A number of FTAM implementations are being utilized, including Marben, Tecsiel and ISODE. It is intended that demonstrations will be carried out using heterogeneous hardware and software platforms to illustrate the interoperability accruing from the use of standards.

The FTAM implementations being used are being enhanced to support access to subunits of a CGM and this is discussed in the next section.

6. STUDY OF SERVICES

The Study of Services workpackage has to date concentrated on two areas where a strong interaction between graphics and networking technologies have been identified:

- (1) the mapping of CGM onto FTAM via a new Document Type, to support the type of application exemplified by the application demonstrator described in section 5;
- (2) the mapping of the X Window System onto a full OSI stack, rather than the mappings available at present, as an aid to making this system available over wide-area OSI networks in particular, and hence promoting its acceptability as a candidate for ISO/IEC standardization.

6.1 MAPPING OF THE CGM ONTO FTAM

In order that a relatively complex structure such as the CGM can be accessed effectively by FTAM, it is necessary to define a mapping of the CGM structure onto the FTAM 'virtual filestore'. There are mechanisms within ISO/IEC for standardization such mappings, which are known as FTAM Document Types, and the work being done here is being fed into this process.

Graphics metafiles provide a mechanism for storing graphical information for purposes such as transfer to another system for display, storing partly developed pictures for later modification and storing pictures for incorporation into documents.

ISO/IEC 8632 'Metafile for the storage and transfer of picture description information' provides a format for capturing 2D static pictures. The format consists of an ordered set of elements, which have a hierarchical structure as shown in Figure 2.

ISO/IEC 8632 is a multipart standard. Part 1 specified the CGM elements, their parameters and the structure of a metafile (constraints on the permitted sequencing of elements). Parts 2 to 4 define these different encodings to represent the abstract syntax of the CGM elements, character encoding, binary encoding and clear text encoding. These have different characteristics relative to ease of generation and interpretation, size of the resulting metafile and ease of transmission between systems.

Several Amendments to CGM are being processed within ISO/IEC. Amendment 1 is of current relevance to this project in that it provides a picture segmentation mechanism. This provides a way to reduce the volume of information required to represent a picture by allowing units which occur repeatedly to be stored only once and invoked by name.

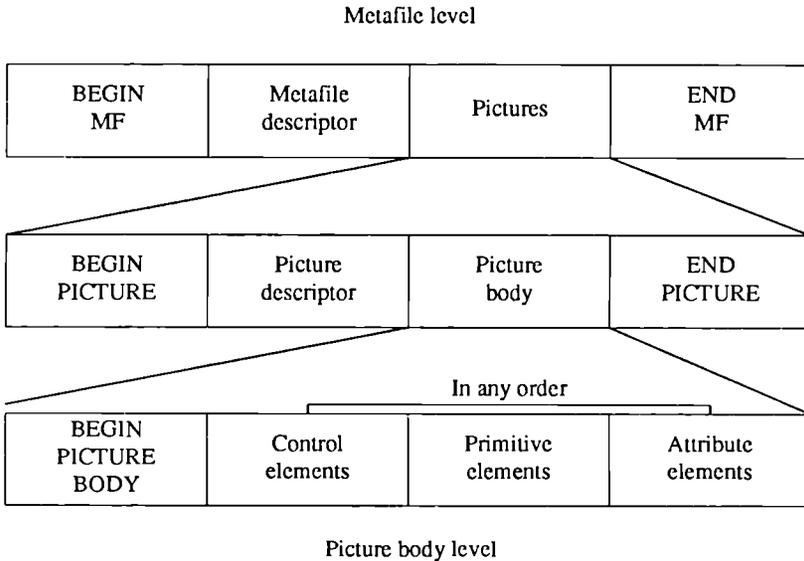


Figure 2: CGM structure

FTAM provides facilities for remote access to a filestore (see Figure 3).

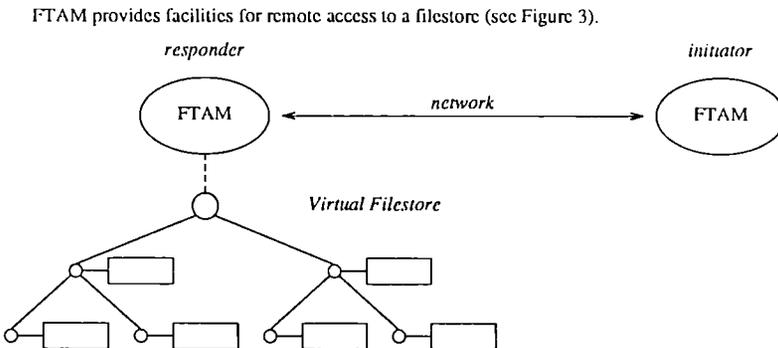


Figure 3: FTAM operation

The ways in which real filestores are implemented vary considerably between real existing systems. Different systems have a wide range of style for describing the storage of data and the means by which it can be accessed. A common model for describing files and their attributes is needed before services, protocols and procedures for file transfer, access and management can be used in an OSI environment. Such a model is provided by the FTAM *virtual filestore*.

FTAM defines a virtual file as 'an unambiguously named collection of structured information having a common set of attributes'. The basic access units of a file and their relationships are described by the file access structure. This is a tree structure that describes the file in terms of the units can be accessed separately (FADUs - File Access Data Units). A FADU is the minimal access unit and is a subtree of the hierarchical file; the actual data units are associated with the nodes of the tree, and can be identified by their relative node name. This is illustrated in Figure 4.

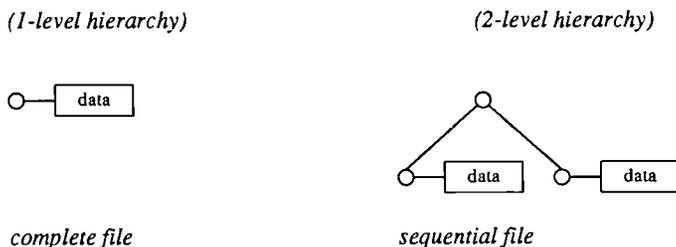


Figure 4: Examples of FTAM virtual filestores

FTAM introduces the idea of a *constraint set*, a set of restrictions and refinements of the general file model, tailored to the needs of a particular class of applications. A *Document Type* defines *inter alia* constraints on the file structure, the data types forming the data units and how the record structure of a real file maps onto the FTAM virtual filestore.

Originally two CGM Document Types were defined within the project, corresponding to the three standardized encodings of CGM (One Document Type for the 'binary' encoding and one to be used with either the 'clear text' or 'character' encodings), in the spirit of other FTAM Document Types. Subsequent work showed that the 'clear text' encoding is unsuitable for structured transfer via FTAM, and that the other two encodings can be handled via a single, parameterized Document Type. The mapping of the CGM structure onto FTAM is illustrated in Figure 5. This structure will allow remote CGMs to be treated by an application in a similar way to local CGMs (ie with knowledge of the structure). Individual pictures within a metafile may be transferred, thus allowing efficient use of bandwidth.

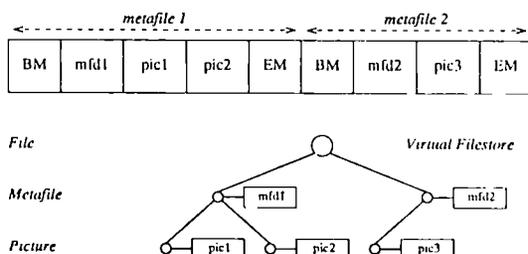


Figure 5: Mapping of CGM onto FTAM Virtual Filestore

The draft CGM Document Type being presented to ISO/IEC does not include support of CGM Amendment 1, but the structure can be extended to support segments by adding further levels of hierarchy. This will be explored in the context of the prototype application. However to do so sensibly requires additional constraints to be placed on the element sequencing defined in CGM in order to eliminate potential side effects of some elements and other dependencies. These constraints are to be investigated and it is envisaged that an Application Profile for CGM will be developed, which embodies these constraints, to allow more structured access to a CGM by FTAM

6.2 MAPPING OF THE X WINDOW ONTO AN OSI STACK

ANSI, the US national standards body, has been working for some time on the creation of a national standard based on the X Window System. It is their intention to submit the national standard to ISO/IEC for 'fast track' standardization. To do this requires the development of an internationally acceptable mapping of X Windows over an OSI network; at present mappings only exist for TCP/IP and DECnet networks.

X Windows provides a windowing system based on the client/server model, and this model can be entered across a network. This is illustrated in Figure 6.

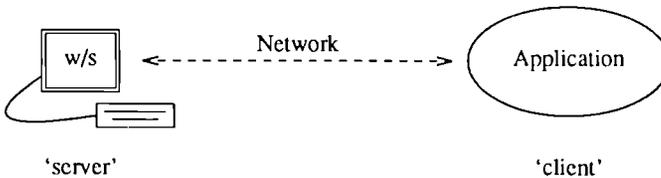


Figure 6: Client/server architecture of X Windows

The specification of X Windows asks only for a 'reliable byte stream' to be provided to do this. This is usually interpreted as a mapping at the Transport Service level. If an attempt is made to map X Windows over a full OSI stack an immediate problem is encountered in that the full stack provides more data-management services than are required by X Windows. Many of these services are either not required, or are already provided by the X Windows *byte stream protocol*. The extent of this overlap is shown in Figure 7.

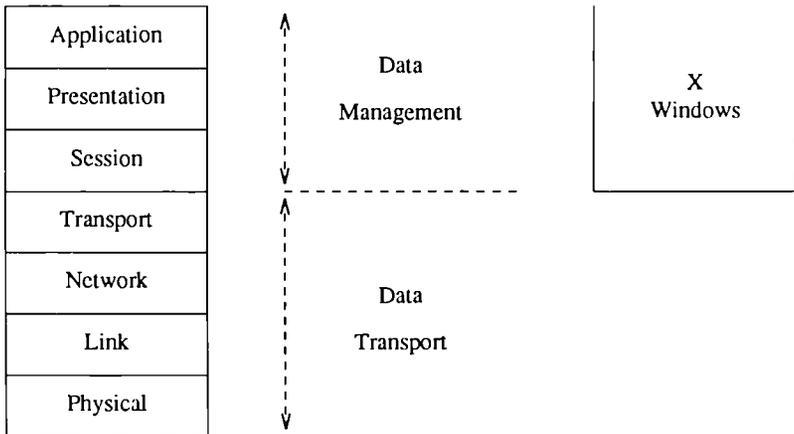


Figure 7: Mapping of X Windows onto an OSI stack

A number of developers have already attempted a mapping of X Windows over the full OSI stack, and over a partial stack extending up to the Transport Layer. Performance measurements have been made of such implementations. The general conclusion is that a mapping over a high-quality OSI Transport Service Implementation gives equivalent performance to a mapping over TCP/IP. This is unsurprising, albeit reassuring. However, mappings over a full OSI stack are found to suffer a performance degradation of one-half to two-thirds of the performance of the TCP/IP equivalent. Some of this degradation can be accounted for by implementation inefficiencies (the measurements have been made largely using prototype versions of the supporting stack), but there has also been found to be a penalty in the mere fact that fully-functional Session and Presentation Services are in use.

These experiments have led to the conclusion within the community that a *lightweight* full-stack implementation is required, supplying *only* those services that X Windows needs. Such an implementation must be efficient whilst retaining conformance to OSI standards, and hence an ability to interwork. The first step is to produce a mapping definition that is carefully constructed to allow a light-weight implementation, but which is aligned to work in progress now within ANSI and later within ISO/IEC so that interworking between implementations is possible.

Such a mapping is being progressed through the auspices of EWOS by various interested parties, mainly although not exclusively in the UK. The project is supplying the effort to edit this document, which has now been completed from the technical point of view and was ratified in May 1991 as an EWOS 'Technical Guide'. The document will also be issued as a CEN Technical Guide, to ensure wider circulation.

In parallel with the development of the text of the mapping, a trial implementation is being made within the UK academic community. Although not yet complete, preliminary results are encouraging. Under the VMS operating system on a VAXstation, a version of DECwindows mapped to this OSI stack has been found to give a performance penalty on throughput in graphical applications of only 20% compared to DEC's own highly-tuned proprietary network (DECNET). It is expected that further improvements will be possible to the implementation, ensuring virtually identical performance over the two networking technologies. This is extremely encouraging, demonstrating that it is possible to build high-performance OSI applications when careful consideration is given to the design issues involved.

The EWOS mapping will be able to be used as an intercept of any forthcoming ISO/IEC standard. When the latter emerges for 'fast track' it is anticipated that ARGOSI will have a role to play in harmonizing European responses to the draft put forward for standardization. In the meantime work is ongoing in studying a number of areas where X Windows standardization is incomplete. Particular attention is being paid to the subject of the establishment of initial server/client connections in the X Windows environment, an area where there is at present no satisfactory OSI mechanism available.

6.3 ASN.1 ENCODING FOR CGM

One of the recommendations of the Breuberg Workshop (see Section 7) was that ARGOSI should develop an ASN.1 encoding for CGM. ASN.1 separates the issue of describing the structure of information, from the issues of how that information is to be represented for transmission. Given an ASN.1 description of a message, a representation can be derived mechanically by applying a set of encoding rules. Such a set of rules, known as the Basic Encoding Rules (BER), has been standardized alongside

ASN.1 ASN.1 and the Basic Encoding Rules are ISO standards 8824:1987 and 8825:1987 respectively. At the present time, CCITT have published agreed revisions to the 1987 ISO standards (Recommendations X.208 and X.209, 1989), but the corresponding ISO documents are still in the publication process.

The main motivation for an ASN.1 CGM encoding is to provide a proper linking between the abstract functionality and syntax of the CGM and the concrete transfer syntax provided through the ASN.1 encoding rules. The benefit of this approach is to bring CGM encoding into the mainstream of OSI development and to take advantage of the tools being provided by OSI developers to manipulate ASN.1 representations. Tools (ASN.1 "compilers") are available to generate programming language code that will generate encodings in the BER from an ASN.1 description of an information structure and an instance of the structure, and to perform the corresponding decoding process. It is also hoped that by using ASN.1, some of the problems experienced currently with transferring CGM binary and character encodings across OSI networks, such as those caused by character translation, can be eliminated.

Two immediate applications for an ASN.1 encoding are in conjunction with FTAM and MHS. The former would enable FTAM implementations to make use of ASN.1 tools for handling a CGM document, rather than two *ad hoc* interpreters, one for each encoding method supported by the implementation. The latter would enable CGM to be registered as a body part for MHS (it is believed that CGM could not be registered without an ASN.1 encoding, albeit the encoding could present the CGM as an unstructured OCTET STRING as a minimum requirement).

The project has agreed a programme of work to develop an ASN.1 encoding for CGM and prototype tools to generate and interpret CGM's following this encoding. These will build on existing ASN.1 and CGM tools available to partners.

The work has been drawn to the attention of ISO/IEC JTC1/SC24, the subcommittee responsible for computer graphics and whilst no decision has yet been made to start an ISO/IEC project to standardize an ASN.1 encoding for CGM, ARGOSI has been given firm encouragement to proceed with the technical work in order to demonstrate the benefits of ASN.1 encodings. SC24 has also decided to consider the implications of ASN.1 for other graphics standards, such as archive files and the CGI. There is a strong possibility that an ASN.1 encoding will be standardized for the proposed image interchange format and ARGOSI will liaise with this activity.

7. GRAPHICS AND COMMUNICATIONS WORKSHOP

The ARGOSI project organised an international workshop on graphics and communications, in conjunction with Eurographics (the European association for computer graphics), which took place in Breuberg, Germany, from 15-17 October 1990. The workshop was attended by 29 participants from 8 countries. After a series of presentations covering a wide range of technical topics, 6 major topics were identified for further discussion.

- Impact of multi-media
- Improvements to the CGM
- The role of profiles
- Relationship to X to standards
- Image Interchange standards
- ARGOSI classification taxonomy

Working groups were formed to discuss the first 3 topics in the above list. The workshop came out with a number of recommendations which have been taken up by the ARGOSI project. The major recommendations were that an ASN.1 encoding of CGM should be developed and the role of profiles for standards for graphics and networking and in particular the applicability of the International Standardized Profile (ISP mechanisms, should be examined.

A report of the workshop has been published,¹ and proceedings are in press.⁴

A second workshop is currently being planned for December 1991. The topic will be Distributed Window Systems - Theory and Practice and will explore the wider issues that has surfaced during the X/OSI standardization activity.

ARGOSI representatives contributed to a workshop on Graphics and Networking in the USA, organised by ACN SIGCOMM and SIGGRAPH in January 1991. The announcement of the workshop was received in October 1990, just prior to the Breuberg workshop. The ARGOSI work was well received at the workshop and the final report noted that:

- "The work reported at the ARGOSI workshop... complements the contents of this workshop continued collaboration between the two communities is recommended."
- "The ARGOSI taxonomy for graphics and networking applications represent a good framework for classifying new visualisation applications."

The workshop provide valuable exposure for the ARGOSI project. It showed that serious work on the integration of graphics and networking was being undertaken in Europe and the European contributions were perhaps those that addressed the subject of the workshop best.

8. CONCLUSIONS

Graphics and networking at enabling technologies finding use in an increasing number of application areas, from weather forecasting to remote consultancy in the medical domain. Standardization is a key issue in graphics and networking in order to promote coherence within the market place and avoid fragmentation. The ARGOSI project is demonstrating that integration and harmonisation between different areas of standardization activities is also of key importance and through the construction of a prototype application demonstration is endeavouring to show the practical benefits such harmonization can bring.

The project is already having a practical impact in the working practices of the industrial partners in the project in terms of the approaches now being taken to the construction of real application system.

For more information, please contact the Project Manager:

J-J Bardyn
 Thomson-CFS
 Division Outils Informatiques
 160 boulevard de Valmy
 92704 COLOMBES Cedex
 France
 Tel +33 1 4760 3407
 Fax +33 1 4760 3636

References

1. D.B. Arnold, D.A. Duce, and D.C. Sutcliffe, "Report on the Eurographics/ARGOSI Workshop on Graphics and Networking," *Computer Graphics Forum* 9 (4)pp. 385-387 (1990)
2. D.A. Duce, J.R. Gallop, G.J. Reynolds, and D.C. Sutcliffe (eds), "Third deliverable: Report on Classification," ESPRIT II Project 2463, ARGOSI (1991)
3. J.R. Gallop, "the ARGOSI Classification Scheme for Graphics and Networking Applications," in *Graphics and Communications*, ed D. B. Arnold et al., Eurographic Seminars, Springer - Verlag O. in press
4. D.B. Arnold et al. (eds), *Graphics and Communications*, Eurographics seminars Springer - Verlag O. in press

AN ADVANCED COPROCESSOR ARCHITECTURE FOR FAST PUBLIC KEY CRYPTOGRAPHY AT VARIABLE KEY LENGTH

Schützeneder Heinz
Siemens AG
Semiconductor Group
Balanstr. 73
P.O. Box 80 17 09
D-8000 Munich 80
Phone : (.89) 4144-3709
Fax-Nr. : (.89) 4144-4071

Eberhard Günter
Siemens AG
Semiconductor Group
Balanstr. 73
P.O. Box 80 17 09
D-8000 Munich 80
Phone : (.89) 4144-2682
Fax-Nr. : (.89) 4144-3683
Telex : 52108-0 sie d

Summary

For EDP the confidentiality and integrity of private messages has to be ensured by authentication means, e.g. digital signatures. Symmetric methods are not appropriate for key distribution over large communication networks. However, Public Key Crypto (PKC)-Systems enable easier key management. With these Asymmetric Algorithms only one of two keys is kept secret.

A Microcontroller and an EEPROM embedded into a Smart Card represents the most secure location preserving this key. Because of the high computational complexity of asymmetric algorithms a dedicated Crypto-Coprocessor was developed. It serves as a High-Speed Arithmetic Unit for Modular Multipli-cations, nearly all PKC-Systems in common. For example, the RSA Algorithm and Zero-Knowledge Schemes are program-able by the Chip Operating System.

The SIEMENS component "SLE44C200" is based on an Advanced Coprocessor Architecture. With 4 cycles of a 140 bit-parallel Arithmetic Unit (AU) it executes one 540-bit RSA-Decryption in about 0.4 sec. at 3.57 MHz. In $1\mu\text{m}^2$ CMOS Technology the Co-pro-cessor's chip area is less than 5 mm^2 . For VLSI-Imple-mentations it is integrated together with a Microprocessor (including 256 Byte RAM), 8 kByte ROM, and 2 kByte EEPROM into one single chip.

By application-specific SW the SLE 44C200 becomes a Multifunctional Processor Chip Card apt for a broad range of applications, e.g. banking card, access control, and PC-Fax in the VANS market. Further flexibility is attributed to a highly regular and modular design of the Crypto-AU. With this property different security levels can be satisfied by an increased key length. A hybrid concept was chosen in order to combine higher baudrates with improved security.

SIEMENS will demonstrate the Coprocessor functionality by means of an Emulator-/Simulator-Tool during the ESPRIT 1991-Exhibition.

Main

Traditional tasks of encryption methods mainly focused on the concealment of confidential messages. However, in the age of Electronic Data Processing new and different requirements on crypto-systems turned out to be very important. Particularly confidential data and information has to be protected against the loss of integrity while authenticity has to be guaranteed. Because of the constantly increased number of users in corresponding electronic systems a secure key management is indispensable. It is

possible to achieve these objectives largely by means of Identification and Authentication schemes.

Conventional techniques, i.e. Symmetric Methods (e.g. D.E.S. or SCA-85), necessitate a secure communication channel for key distribution. That has to be taken into account with the realization of these systems and leads to rather complex configurations. By the use of Public Key Crypto-Systems this can be overcome and advanced security requirements can be satisfied. Because of the high computational complexity of these so called Asymmetric Algorithms, SW-Realizations are excluded in advance from the performance point of view. By contrast, VLSI-Implementations are most suitable concerning the attainable effectiveness since here concurrent processing can be applied most flexibly. But also by means of a Cryptography Processor which is composed of commercially available components, as e.g. standard Digital Signal Processors, the required calculation speed will not be achieved economically. Similarly, multi-chip solutions suffer from the necessity of additional physical security measures for the safety of key management. At present, the convenient features of public key crypto-systems are bought at the expense of speed. Therefore, an efficient HW-Implementation through a Single-Chip Solution is the primary goal in order to avoid the limitations. A non-volatile memory part (EEPROM) serves as a security memory, e.g. for the storage of secret keys. The keys are best preserved when the corresponding security memory is embedded in a Chip Card-IC, in which all memory accesses are controlled by a microprocessor via the Chip Operating System.

Nevertheless, especially in Smart Card Implementations the realization of asymmetric algorithms had been a problem with regard to both, memory requirements and performance. For instance a RSA Decryption procedure (all 512-bit operands) takes several minutes with today's available smart card technologies. In order to decrease calculation time drastically (1 sec.), corresponding to the major applications, namely RSA-based message and user authentication, a High-Speed Arithmetic Unit for the calculation operation in question, i.e. Modular Exponentiation (exponentiation over a bicomposite product of primes), is needed. Finally, an efficient CRYPTO-COPROCESSOR for fast performing of Modular Multiplications is necessary. This COPROCESSOR should also be in conformity with the demands of further known asymmetric algorithms based on modular multiplications, as e.g. the Zero-Knowledge Proof Protocols according to FIAT-SHAMIR and GUILLOU-QUISQUATER.

The objectives will be achieved with the next generation chip cards, such as the one announced by SIEMENS (SLE 44C200) and supported by the Council of the European Community in the ESPRIT Project EP 2704 "CRYPTO IC - CARD".

The goal to improve remarkably the performance of asymmetric authentication schemes for smart cards without decreasing their security will be reached by an Advanced COPROCESSOR Architecture. It uses a Dedicated 140 Bit-Parallel Arithmetic Unit, implemented the first time in Europe on a monolithic chip card IC. The COPROCESSOR of SIEMENS executes the modular exponentiation operation " $C = M^E \bmod N$ " (all 540-bit words) in about 0.4 sec. (at 3.579 MHz clock rate according to ISO). That means a gain in performance by at least a factor of 6 compared to other conventional solutions. Since the chip area amounts to less than 5 mm^2 ($1 \mu\text{m}$ CMOS-Technology) the Coprocessor is best suited in view of chip card applications.

The advanced coprocessor architecture utilizes multiple Look-Ahead (LA-) Algorithms to speed up the time for the modular multiplication operation. Three LA-Algorithms are implemented for acceleration of the multiplication, division (and hence the modulo-operation) as well as exponentiation. The values of the latter LA are derived by means of the anticipated Controller-SW, whereas the next parameters (for shifting and

adding) pertaining to the two LAs for modular multiplication are calculated in parallel by additional HW. Since these values are computed simultaneously to the operation of the adder circuits, the critical path could be shortened substantially. Analogous to the multiplication based on additions, the modulo-operation is reduced to iterative subtractions of the modulus using the binary two's complement method. As a 3-operand adder is provided for computing the basic addition operation, the subtraction can be performed concurrently by means of the chosen adder architecture.

The following full adder, which merges the intermediate 2-bit result of the previous adder stage into a 1-bit sum, is supported by a sophisticated Carry-Look-Ahead (CLA) logic. The corresponding row of full adders is divided in a chain of 20-bit sized adder elements, called block adders. Thereby, each block adder has an attached block CLA element. Both units together are generating the block carry bit as well as are calculating the 20-bit result, respectively. Therefore, it is possible to perform one multiplication step requiring an addition and one modulo-operation step necessitating a subtraction in a single processor cycle. When a block carry bit depends on the corresponding bit of the next lower block adder (probability merely 2^{-20}) an interrupt circuitry (activated by this bit) disconnects for a certain amount of cycles the triggering clock lines of the Crypto Arithmetic Unit (Crypto-AU). Thereby, the adder becomes self-timing and asynchronous. The control logic is used to close up the unnecessary waiting gaps between adjacent arithmetic operations, because most additions require much less time than that of the worst-case delay. The major advantage of this proceeding lies in the fact that the throughput rate of addition is determined solely by the average addition time rather than by the longest lasting addition.

Additionally, the architecture is adjusted to the requirements specific to Chinese Remainder Theorem, e.g. applicable to decreasing the crucial generation time for Digital Signatures.

Since the wordlength of the Crypto-AU is a multiple of 20 bit, a size of 140 bit is considered to represent an optimal compromise between the attainable calculation speed on the one hand and the demand on Silicon area on the other hand. With the chosen concept a modular multiplication employs 4 consecutively cycles within the Crypto-AU in order to process 540 bit (20 bit needed for overflow) words. The 140 bit RISC-type Coprocessor Architecture together with the corresponding Controller Circuits and the available Coprocessor RAM resources of 5-times 540 bit (working and I/O-registers) result in a chip area of below 5 mm^2 ($1\mu\text{m}$ CMOS-Technology). This remarkably small chip area was achieved by full-custom bitslice-oriented design style which leads to highest regularity and packaging density of transistors (up to 10,000 [trans./ mm^2]) ! On account of this small die size and the achievable calculation speed the new COPROCESSOR is best suited for VLSI-Implementations with regard to Smart Card applications.

Until now the too small computing power of chip cards pertaining asymmetric crypto-systems prevented their use or diminished at least the attainable security level. The out-standing Crypto-Coprocessor performance makes it possible to apply a Smart Card unrestricted for Asymmetric Authentication Schemes. This is a real innovation! Therefore, this key element opens the door for a new class of card applications in the security chip card market. Fig. 1 of the Appendix is illustrating its relevance and wide applicability.

Furthermore, the architectural concept is tailored to a modular design so that the size of the resultant circuit can easily be expanded by additional HW without changing the Controller Circuit significantly. With this expandable feature the security of crypto-systems can be improved to the users' desired level by simply increasing the key length.

The currently implemented 540 bit are considered to be sufficient in terms of most of today's applications. With regard to future higher security requirements two promising alternatives concerning the Key Length Modification Impact on performance and chip size have been investigated by SIEMENS.

A feasibility study for an increased key length of 700 bit as well as 1080 bit was carried out in order to show the influence of greater key sizes on performance, chip resources, and expenditures (in terms of arising costs and design effort). The study revealed that both versions could be realized with an estimated growth of Silicon area of 20 % (700-bit key) or 70 % (1080-bit key), respectively. The corresponding diagram "Relative Chip Area Demand versus Key Length" is shown in Fig. 2 of the Appendix.

Using the 140 bit AU the cycles of the Crypto-AU performing one 1080-bit modular multiplication would be doubled, i.e. 8 cycles are needed. In combination with the doubled number of multiplier and exponent bits this finally amounts to a decryption time (all 1080-bit words, at 3.579 MHz clock) of 3.2 sec., cf. Fig. 3 of the Appendix. With future submicron technologies SIEMENS could even implement a wordlength of 280 bit for the Crypto-AU, resulting in an estimated decryption time of less than 2 seconds for the same 1080 bit key length.

Since there is a tradeoff between calculation speed, die size, and security level a 700 bit key length - requiring a 180 bit Crypto-AU - seems to represent the optimal compromise. An analysis for optimizing chip resources, calculation speed and security levels, aimed at rising security requirements for true electronic signatures in the future, is shown in Fig. 4 of the Appendix. Due to the 40-bit increased Crypto-AU the continuation with the advantageous four-cycle concept is possible. A decryption procedure could then be calculated in about 0.84 sec. (all 700-bit operands with 3.579 MHz clock), see Fig. 3 of the Appendix.

A summary of the Crypto-Coprocessor chip performance features at several key lengths configurations is listed in Table 1 of the Appendix.

Not only this expandable property concerning the key length but also the fact that the proposed circuit is SW-programmable offers a great deal of flexibility! An adapted coprocessor instruction set allows the performing of almost all currently known algorithms based on modular multiplications. Beyond the RSA Algorithm the Zero-Knowledge (ZK) Identification and Signature Schemes of FIAT-SHAMIR and GUILLOU-QUISQUATER are supported by the Coprocessor. Since the Coprocessor provides the computational power necessary, as well as enough RAM (350 byte) for intermediate data storage, the specific requirements of ZK-Proof Protocols are fulfilled by a combination with EEPROM memory. This alterable memory adjacent to the 8-bit microcontroller is recording transactions between prover and verifier and serves as a security memory for the secret keys and parameters necessary. Also the wide-spread 8051 instruction set of the 8-bit microprocessor ensures efficient SW-development for implementing the various algorithms.

Although the implementations of asymmetric smart card authentication schemes run at some thousands bits per second in the coprocessor, the symmetric method, i.e. D.E.S. or SCA-85 (SICRYPT algorithm with essential higher baudrates), is also supported in the Crypto IC-Card by means of software.

In order to increase performance for conventional en-ryption and decryption tasks, it is generally desirable to make use of a hybrid concept. Therefore, the most important application of public key systems is their use for secure key distribution. As a consequence the concept of a certificate - a cryptographically authenticated message containing a cryptographic key - plays a vital role in modern key management.

As the generation/verification of the digital electr. signature (the most promising security mechanism of the future) includes message authentication, the corresponding figure of the Coprocessor has to be pointed out. A generation and verification of a digital signature takes less than 0.1 sec. (all 540-bit operands, 3.579 MHz clock) by utilizing the Chinese Remainder Theorem, as shown in Fig. 5 of the Appendix. By applying the en-/decryption protocol simultaneously, an authenticated signed secret key can be transported from a user to his communication partner. After having received the secret key the establishment of a secure end-to-end communication is then based on any other fast (symmetric) cipher.

According to the Project Time Schedule the Coprocessor will be integrated in an advanced 8-bit Chip-Card Micro-controller with 8 kByte ROM, 2 kByte EEPROM, and 256 byte RAM, embedded in Chip Cards and made available to the market.

The basic algorithm for the fast performing of modulo multiplications, used in the coprocessor, as well as the estimated performance was verified successfully by means of computer simulations.

A further result is referred to the Emulator Concept demonstrated in the ESPRIT 1991-Exhibition. It serves as an important tool for practical applications in new systems.

OBJECTS + SCRIPTS = APPLICATIONS

Oscar Nierstrasz
Dennis Tsichritzis
Vicki de Mey
Marc Stadelmann
Université de Genève
Centre Universitaire d'Informatique
12 rue du Lac, CH-1207 Geneva, Switzerland.
E-mail: {oscar,dt,vicki,marc}@cui.unige.ch.
Tel: +41 (22) 787.65.80. Fax: +41 (22) 735.39.05.

Abstract

We argue that object-oriented *programming* is only half of the story. Flexible, configurable applications can be viewed as collections of reusable objects conforming to standard interfaces together with *scripts* that bind these objects together to perform certain tasks. Scripting encourages a *component-oriented* approach to application development in which frameworks of reusable components (objects and scripts) are carefully engineered in an evolutionary software life-cycle, with the ultimate goal of supporting application construction largely from these interchangeable, prefabricated components. The activity of constructing the running application is supported by a *visual scripting tool* that replaces the textual paradigm of programming with a visual paradigm of direct manipulation and editing of both application and user interface components. We present scripting by means of some simple examples, and we describe a prototype of a visual scripting tool, called *Vista*. We conclude with some observations on the environmental support needed to support a component-oriented software life-cycle, using as a specific example the application development environment of *ITHACA*, a large European project of which *Vista* is a part.

1. Introduction

Although object-oriented technologies are gradually being recognized as a critical contribution to alleviating the "software crisis", it is also apparent that it can be quite difficult to achieve more than marginal benefits simply by switching to object-oriented programming languages. An arbitrary application may be just as difficult to program with an object-oriented language as with a standard block-structured imperative language—perhaps even more so—and the principal benefits of such an exercise, namely a cleaner and more manageable decomposition of the code, are highly sensitive to the results of the object-oriented analysis and design phases. There is no guarantee that any of the software developed will be truly reusable or that it will survive radical changes in requirements. There is not even any guarantee that an application developer will be able to re-use any non-trivial, previously developed classes in the implementation of a new application.

We claim that this is so because object-oriented languages are still largely perceived as a *programming* technology rather than as a software component production technology. This, in turn, is because we still largely approach software development as

a labour-intensive craft rather than as a capital-intensive engineering discipline [5][22][26]. Rather than expending all our effort and creativity on developing individual applications, we should be investing more effort and creativity into the development of standard components, interfaces and tools. The vast majority of applications should be seen as customized assemblages of largely standard parts. Such a scenario, however, requires a shift in our attitudes from short-term, single project software life-cycles to long-term, evolutionary life-cycles that accommodate frameworks that will be used for many projects to come. A software industrial revolution necessarily entails a revolution in the *economics* of software production. We shall not concern ourselves here with the problem of how to bring about such a revolution; we shall confine ourselves to some of the technical problems involved.

We would like to be able to use object-oriented technology as a means to realizing our scenario for component-oriented software development. There are three main technical obstacles:

1. Top-down strategies for requirements analysis, specification and design are unlikely to arrive at any reusable components at the implementation stage.
2. Object-oriented languages typically provide a very limited binding technology for composing software at the level of expressions and statements [5]. In order to compose existing objects one must *program* some new objects.
3. Software composition is made *more* difficult by the ability to define rich interfaces to objects. The most successful examples of reusable components rely on the existence of fairly simple, standard interfaces [27].

The first problem is essentially a methodological one, but is directly addressed by the development of frameworks [27], which package standard software solutions in terms of collections of composable software parts. It is helpful to view application development as a *sales* activity, in which the developer is *negotiating* to sell an available, customizable product, rather than trying to build an entirely new product. Frameworks, then, correspond to product lines, which are developed by R&D departments, not sales departments. By separating these two activities, product reliability can be improved and costs lowered.

To address the second problem, we propose *scripting* as a binding technology for object-oriented languages: a script introduces and binds a set of objects-or, more generally, a set of software components - that will then collaborate to solve a particular problem.

The third problem is addressed by the notion of a *scripting model*, which defines a set of standard interfaces for the components of a particular framework and specifies which components are plug-compatible and how exactly the binding is achieved. For example, a user interface scripting model is typically event-based, and defines what kind of user interface components exist and how they may be attached to applications. A scripting model for Unix commands and files, on the other hand, is stream-based, and defines how sources, filters and sinks may be "piped" together. New components developed for an existing framework must conform to its scripting model or their reusability will be impaired (as is the case with Unix commands that do not use the standard input or output streams). Scripts simply specify the binding between such components.

Just as a successful business requires an infrastructure and a support environment for marketing and sales, component-oriented software development can only succeed if the proper tools for managing, finding and composing software are available [13]. A *visual scripting tool* is an interactive tool for composing and editing visually presented software components. It supports an application developer in the task of constructing both the internal behaviour and the user interface of a running application from standard software components.

In section 2 we shall introduce scripts syntactically and discuss their key properties. In section 3 we shall illustrate the paradigm of scripting by means of an example from the domain of office systems. We shall then discuss in more detail the notion of scripting models and how they assign a semantic interpretation to scripts. In section 5 we will describe *Vista*, a prototype of a visual scripting tool and in section 6 we will briefly discuss some related work. Finally, in section 7, we shall discuss the role of *Vista* within the application development environment of *ITHACA*, a large Technology Integration Project of the European Community's Esprit programme.

2. Scripts

Scripts provide the "glue" that binds together plug-compatible software components. A software component is any piece of software that imports or exports services. Objects are software components because they both export services - namely their visible operations - and import them, from external "acquaintances" of which they are clients. Classes are also software components, as they export the service of being able to stamp out object instances and they import services from their superclasses. Components are interesting for software composition (1) if the binding of services to clients is delayed, and (2) if the available services are standardized, i.e., if "plug-compatibility" between clients and services is defined. These two properties can then be exploited in a script that specifies bindings between plug-compatible components.

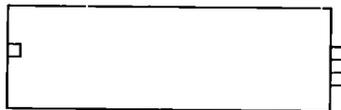


Fig. 1: A component with one input and two output ports

For the moment we can consider scripts as purely syntactic entities. Similarly, we need know nothing more about the services associated to a component than the names by which they may be bound, that is, the component's *ports*. There are *input ports*, each representing a set of available services or properties (i.e., where requests can be



Fig. 2: Linking components in a script

received), and *output ports* representing services required from other components (i.e., from which requests will be issued). We will use a graphical notation in which components are represented by rectangles and input and output ports by squares on the perimeter of a component, respectively inside or outside the rectangle (see Figure 1)

A script specifies bindings between a set of components. In Figure 2 we see this represented graphically by links between the ports of components. A link between the output port of one component and an input port of another means that the first component can use services provided by the second.

Aside from the restriction that output ports can only be connected to input ports, we have not yet introduced any constraints over the syntax of the scripts we can compose in this way. Are there different kinds of ports and links? May we connect multiple inputs to outputs, or vice-versa? Can components have any number of input or output ports? May the graph of a script contain cycles? These questions cannot be answered in absolute terms, but must be considered on the basis of the kinds of ports and links needed for a particular component framework. Scriptable components conform to a *scripting model* (section 4) that formalizes both the syntax of scripts and their interpretation. Let us consider the kinds of syntactic rules we would like to impose on scripts:

One may define different *types* of ports, each representing a different set of services. Only *compatible* input and output ports may be linked. Since one type of port may represent a *subset* of services provided by another, standard polymorphism rules may apply.

One may *combine* ports in order to combine sets of services. It is also possible to combine input and output ports, in which case linking would imply a two-way client/server relationship (an example will be given later).

Since input ports represent availability of services, they may normally be linked to multiple output ports, that is, to multiple clients. Ports that permit multiple links are called *multi-ports*. Certain services, however, will only work for a unique client, in which case one may restrict an input port to be *singular*. Since output ports represent the use of service, they will often be singular. Output multi-ports can be very useful, however, for components that iterate over a set of service providers.

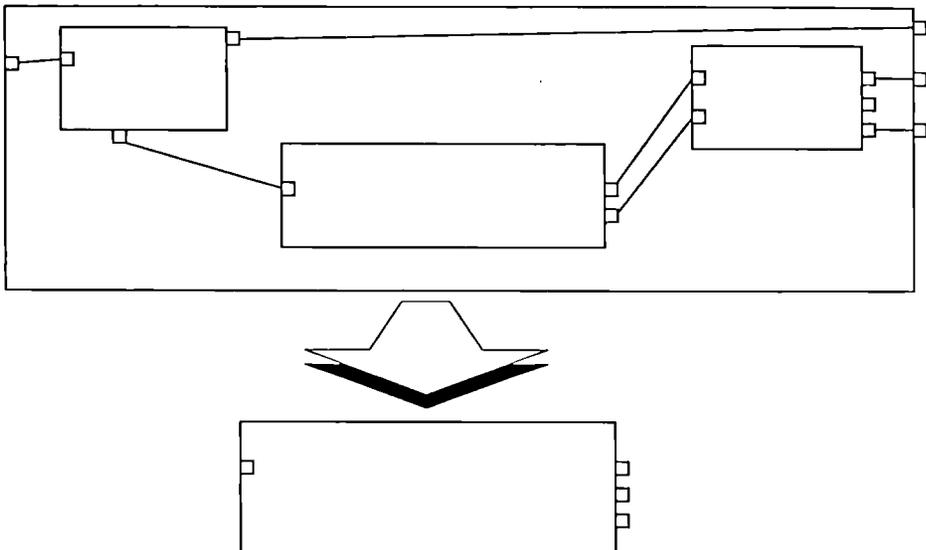


Fig. 3: Encapsulating a script as a component

Binding of ports may be either *optional* or *obligatory*. Output ports typically must be bound to complete a component's behaviour, whereas input ports generally represent availability of services, so may be left unbound. One may also wish to associate *default bindings* for ports.

Certain linking constraints cannot be enforced by considering purely local connections. Such constraints are typically expressed by forbidding certain kinds of cycles in the graph of a script. Such cycles may imply the possibility of a deadlock (e.g., two office procedures competing for the same set of electronic forms), an infinite execution loop (e.g., inside a spreadsheet), or a recursive containment relationship (e.g., a document containing itself). In other cases, however, cycles may be explicitly required, e.g., by a scripting model that supports bidirectional client/server relationships.

There are two further crucial properties of scripts. The first is that *scripts are also components*. One may encapsulate a script as a component in order to reuse it in future scripts (Figure 3). A script as a component encapsulates a set of partially bound components and determines which ports will be visible to surrounding scripts. Note that the "import/export" links of an encapsulated script serve only to rename ports, not to bind them—input ports of the script are linked to input ports of its components, and similarly for output ports. A script as component may also contain "holes" representing components to be filled in later (in this case the port associated with such a hole represents *all* the services of the missing component). Container objects are good examples (not just queues, stacks and bags, but also mailboxes, folders and spreadsheets).

The second important property is that scripts (and components) may have *multiple views*. Not only can one choose between the encapsulated and the expanded views of scripts, but one is free to vary the way in which different types of components, ports and links are visually presented. This is the key to a visual scripting tool which supports the interactive specification and interpretation of scripts: components best represented as graphs can be seen as graphs, but components that have a "natural visualization",

Order Form

Customer: #

Address:

Product	Quantity	Price/Item	Subtotal
<input style="width: 90%;" type="text"/>			
<input style="width: 90%;" type="text"/>			
<input style="width: 90%;" type="text"/>			

Approved: Total:

Fig. 4: An active form

such as electronic forms or user interface components, can be viewed as such. Similarly "linking" can be thought of as connecting components with lines or arrows, but it may also be viewed as gluing or positioning components on a surface, plugging together pieces with compatible interfaces or simply selecting appropriate values from a menu or a property sheet. By properly exploiting the potential for multiple views of scripts, we can (1) support the *direct manipulation* of applications, and (2) *control complexity* by reducing the need for textual names and by selectively displaying only those aspects of an application of current interest.

3. Scripting Active Forms

In a sense, all programming languages obey some scripting model, but typically the level of scripting is very low: only expressions and statements can be plugged together to form new expressions and statements. Scripting, as opposed to programming, becomes interesting the moment one identifies a well-understood application domain in which many applications have similar characteristics that can be factored out as software components. The area of Office Information Systems contains many suitable examples [18]. In fact, electronic spreadsheets, hypertext systems and Fourth Generation Languages can all be seen as examples of scripting tools for specialized application domains.

A prime example of a suitable scripting domain is that of electronic forms. Forms are pervasive in office systems, are well-understood, and are readily decomposed into standard components. As an example, we shall look at how one may script *active forms*. A passive form differs from an active form in that the former functions purely as medium for storing structured information whereas the latter has an associated *behaviour*. For example, by filling in a field of an active form, one may cause a side effect in another field. We make this distinction because we wish to emphasize that scripting is useful not only for building user interfaces, such as the appearance of passive forms, but for constructing the underlying applications as well, such as the associated behaviour of an active form.

Consider the order form shown in Figure 4. At the very least we would like to be able to script the appearance of such a form. At this level we can identify three different kinds of components: form *surfaces*, constant *labels* and modifiable *fields*. Scripting in this sense is essentially a matter of gluing labels and fields to a surface. A form is a surface that *owns* a number of labels and fields, each of which has some associated position on the surface. The scripting model is very simple so far, in that the *owns* relationship is the only kind of link defined, and implies nothing more than the ability of the surface to display its parts. (The actual positioning of the parts on the surface may be static, or may be dynamically determined by means of graphical constraints.) Fields come with their own behaviour that allows users to interact with them.

There are two behavioural aspects of active forms that we would like to script in addition to their appearance. First, fields have an associated behaviour which can be quite rich [12] [24]. Second, forms themselves may support different views depending either on their internal state or on the context in which they appear (e.g., explicit user requests to present a different view, or implicit constraints determined by the capabilities of the current user).

Let us first consider the various kinds of fields present: the # field is a unique identifier for the form which is initialized at form creation time but which may not be modified. The **customer** and **address** fields are plain, modifiable text fields. (Though in a more realistic example, they would likely serve as an interface to a database of customer

records.) The **product** and **quantity** fields are also modifiable text and integer fields, but they should generate (1) a query that retrieves the price per item and (2) a computation that produces the subtotal. Presumably one may order more than just three items, which implies that the list of products ordered is actually a *subcomponent* of the form, a container for a sequence of product/quantity/price/subtotal entries, each of which can be seen as a subform. The **total** field is an unmodifiable integer field whose computation is triggered by modifications in the list of order entries. Finally, checking the **approval** box means that the form can no longer be modified, that is, it implies a side-effect on the form causing it to become a "frozen" view of its contents.

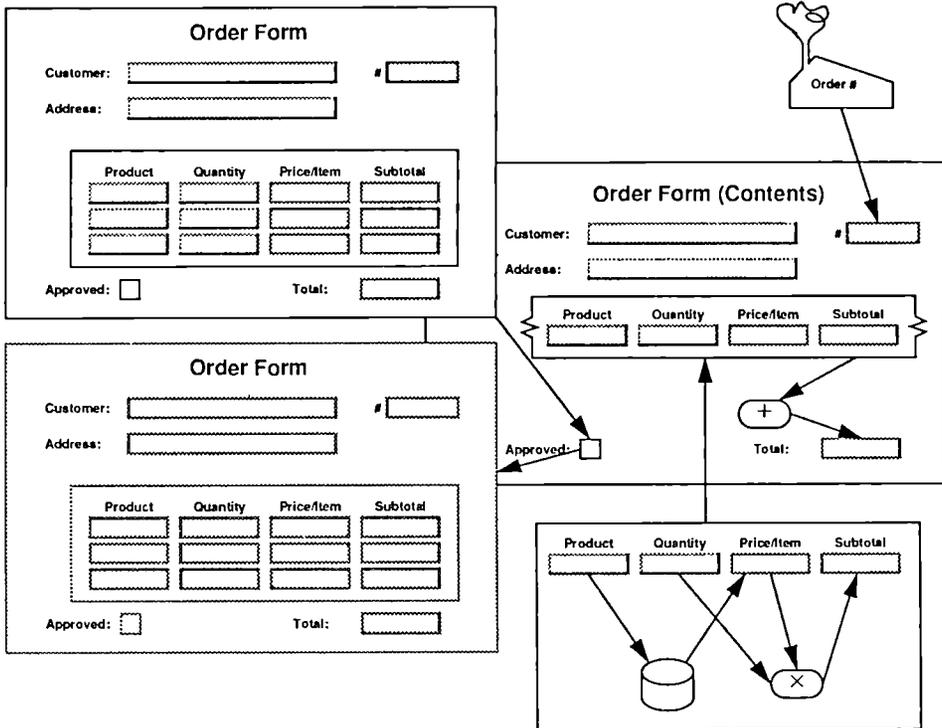


Fig. 5: Scripting an active form

We show how this behaviour might be scripted in Figure 5. The order form script consists of two surfaces visible to the end user representing the unfrozen and frozen views of the form, a contents surface where the field behaviour is scripted, and two external components, the "factory" for unique identifiers, and the product entry script. The fields of the two visible surfaces are essentially views on the underlying contents fields (links are implicit). The view fields of the first surface will forward modifications to the underlying contents, but those of the second surface are all "frozen" and ignore attempts to modify values. The approval field triggers the transition between from the unfrozen surface to the frozen one. The behaviour of order entries is separately scripted and is linked to the container for order entries in the contents surface.

Although we have not explained in detail the kinds of ports, links and components of this informal example, the principle ideas of scripting should be apparent: visualiza-

tion and direct manipulation of components, scripting of both user interface and behaviour, standard kinds of ports and links, multiple views, and using scripts as components.

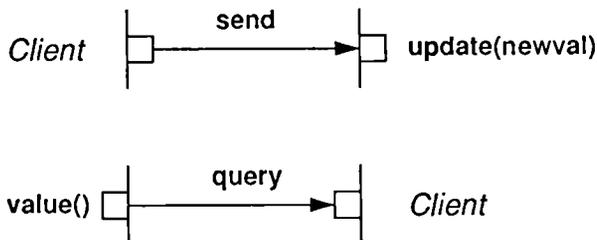
4. Scripting Models

So far we have concentrated on essentially *syntactic* aspects of scripts: their visual representations and rules for composing them. The interpretation of scripts, that is their *semantics*, is equally important, as this is where the connection between scripts and objects is established. Both syntax and semantics must be formalized in the scripting model for a component framework.

The scripting model also acts as a bridge between the application developer and the component developer. Links have, in effect, *two* semantics: one for the interpretation of the script, and another for the components that make use of these links. Consider, for example, the link between a view field on a visible surface of an active form and the corresponding text field of the form contents. To interpret such a link, it suffices to "introduce" the view field to the text field that it is connected to. There are many possible ways such a connection may be established, one of which is that the view field be an object with a "connect" operation that can be called when the connection is made. A scripting tool can guarantee that this connection only be made between plug-compatible fields. In this case, presumably, the view field expects the component to which it is connected to have operations that allow its value to be retrieved and to be updated. Note, however, that the interpretation of the script is strictly limited to managing the connections. It is the business of the component itself to decide when and how to make use of the connection. A scripting tool, then, functions like a corporate lawyer setting up standard contracts between standard kinds of customers and clients. It oversees and authorizes the signing of the contracts, but it is not concerned with their execution.

Scripts may have multiple interpretations depending on how they are to be used. Although the interpretation of a script is limited to establishing connections, this may be done in many different ways. For example, if an active form designed by scripting is to be directly interpreted, this implies that the binding between components be very loose. If, on the other hand, we wish to generate an object class from an active form script, then we might "compile away" the bindings to obtain a more efficient implementation. Differences in interpretation may also be due to different visualization requirements, for example, whether the script is to be executed as an end-user would see it, or in a special "debug" mode that exposes its implementation.

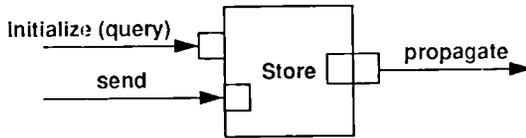
A basic scripting model for active forms is quite simple, requiring only two kinds of links in addition to the *owns* link: a *send* link, which allows one field to update another, and a *query* link, which allows a field to obtain the current value associated with another. We can visualize these links as follows:



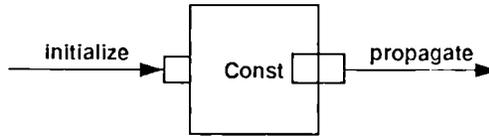
In both of these figures the *client* is the component with the output port and the service is provided by the component with the input port. The services provided are the **update()** and **value()** operations. The *direction* of the links is intended to indicate the direction of information flow, not which component initiates the exchange of information.

The *meaning* of these connections is that the client component will be able to use the indicated operations of the other component it is connected to. Note that in both cases the client is free to use the connection at any time. The scripting model only states what a connection entails, not when it will be used.

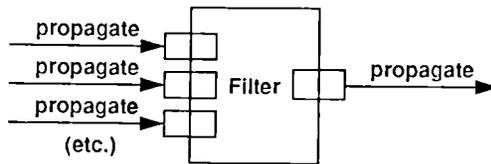
From these basic types of ports and links we can generate some of the others we will need. A store is a kind of component that initializes itself by a query link, can be updated, and can propagate its value to other components. An *initialize* link is a query link that will be used only once, when a component is instantiated. Note that it is the store that *requests* initialization, hence an output port is required. The send link allows the store to be updated. The *propagate* link is a combination of a send link *and* a query link, and so combines an input port and an output port. The propagate link is used by the component to update any connected components whenever it changes state, and it also allows the connected components to query the current state of the store at any time.



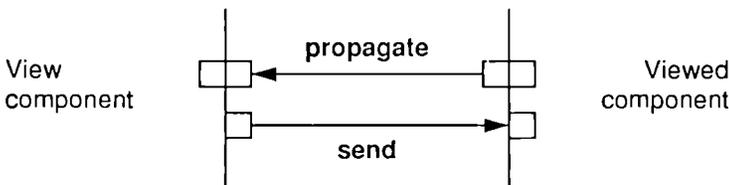
A *constant* is even simpler, since it cannot be updated:



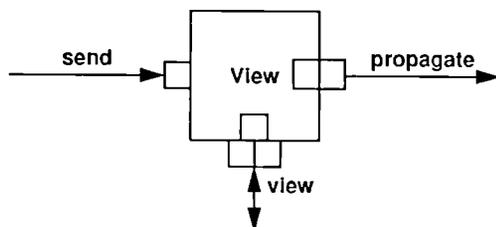
A *filter* is a component with no memory that computes a value from a number of parameters. It can be used either on a demand basis, or by triggering (hence it supports propagate links rather than just query links):



Finally, a *view* field makes use of another kind of composite link called a *view* link. The view component can both ask for the current value of the viewed component and



be notified of updates. (This combination is simply a propagate link towards the view.) In addition, The view component will pass on any updates it receives from other sources. A view component then, accepts updates (from the user or another component), synchronizes with another component by means of a view link, and propagates any changes onward.



With these generic component types, we can classify most of the fields we encountered in our active form example. The customer and address fields are stores and the identifier is a constant. The arithmetic functions and the fields that display them are filters. The fields of the view surfaces are view components (in the case of modifiable fields) and filters (the frozen ones).

The missing links belong to the approval field, which must be capable of communicating between surfaces, and to the order entry container, which must know the kinds of components it has to manage. Each of these links represents the binding of another set of services, which we will not describe here.

5. Vista

We have implemented a prototype of a visual scripting tool for object-oriented applications, called *Vista* [7][8][19]. *Vista* is an object-oriented successor to *VST*, an earlier prototype based on a Unix scripting model [23]. The current version of *Vista* supports the selection of components by means of a menu, the graphical linking of components, interactive execution of scripts, saving of scripts and the use of scripts as components within other scripts.

A small set of components has been implemented for demonstration purposes: text fields, labels, toggle and arrow buttons, slider, system date, "orchestra", simple arithmetic calculator, simple database of records and document. The scripting model presently supported is essentially a dataflow model: components make values available on their output ports and these values propagate to the input ports of connected components. In effect, the only kind of link presently available is the *send* link described earlier. In general, though, links may represent arbitrary sets of services associated with a particular scripting model. In these cases we plan to use the built-in dataflow links as a means to distribute access to services amongst components. The job of *Vista* will be to keep track of which connections are permissible and to actually establish the connections at the level of the underlying objects.

5.1. A Vista example

We shall illustrate the functionality of *Vista* with an example of a simple office procedure. The example can be stated as follows:

There exists a database of customer records of the format *name:fax number:value*, which supports queries and insertions. Given a customer name, the script must generate a fax to the customer containing the date, the customer's name, fax number and a balance calculated from the value in the customer's record and an input value.

Figure 6 shows an implementation of this office procedure as a Vista script. In this case Vista's built-in dataflow scripting model has been used to script the desired behaviour. Simple arrows represent links and component ports are only visible during the linking operation. All links are displayed attached to the upper left hand corners of their linked components. Each link is a dataflow link and the script as a whole can be read as a dataflow diagram.

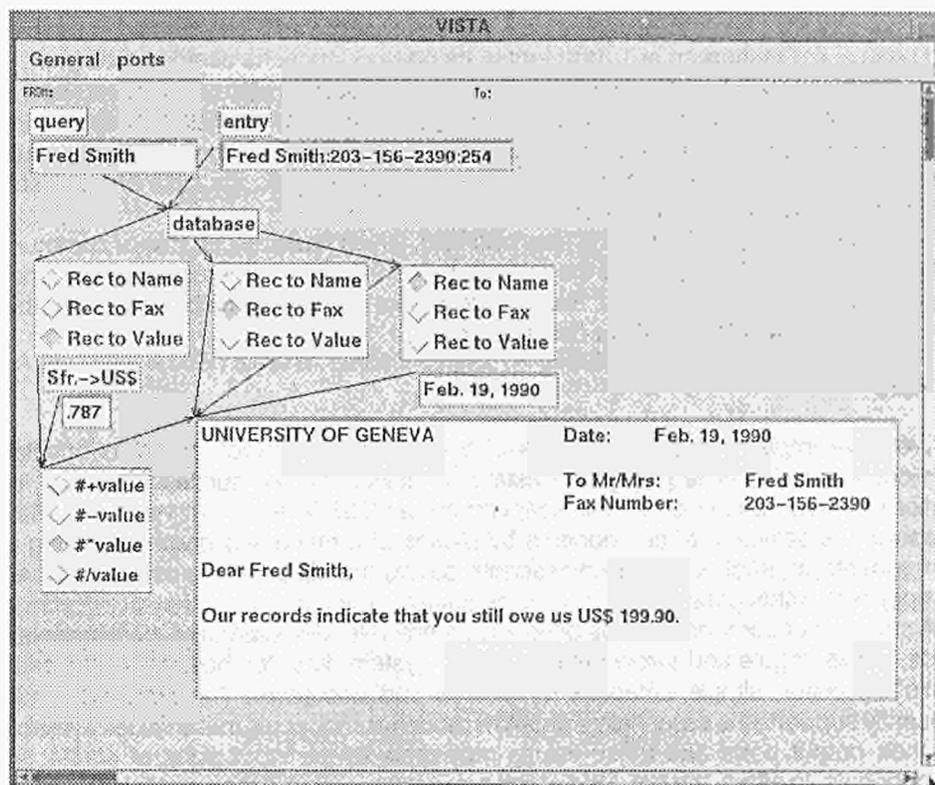


Fig. 6: A Vista Script

Six kinds of standard components have been used in the example: *Label* components, like **query**, **entry** and **Sfr.-> US\$** which simply display a constant text label, *Text* components into which a user may type, a *Database* component that accepts queries and new entries, *RecordTo* components that extract fields from records, a *Calculator* component and a *Document* component that presents the assembled information.

The first two *Label* components serve to document two *Text* components which feed values to the *Database* component. The job of a *Text* component is to store a string. The current text stored in the component is displayed in a rectangle where it may be

edited by the user. In addition, each Text component has an input port called **text in** where it may receive updates and an output port called **text out** which is used to propagate the current value to other components. In this script the **text out** ports are linked to the Database component and the **text in** ports are left unbound.

The Database component is responsible for maintaining a set of records with the format *name:fax number:value*. It has two input ports, **query** and **entry**, where a string to match or a new record to enter may be received, and a **record out** output port where the last record stored or retrieved is made available to other components. In the script this output value is propagated to three RecordTo components each of which extracts and outputs a selected field of the record.

A Calculator component has two input ports for its arguments and a single **result** output port for the value computed. The Document component is used to group and format information produced by its connected components into a fax message that can be sent to the customer.

It should be noted that, in the current version of Vista, there is no difference between creating a script and executing a script. For small scripts this may be acceptable but for larger and more complex scripts, separate creation and execution modes could be useful. Furthermore, it should be possible to generate a stand-alone application from a script-presently scripts can only be executed from within Vista.

5.2 Vista Implementation

VISTA's internal representation of a script is a graph. The nodes of this graph represent components and the arcs represent links. This basic graph structure has no inherent syntax or semantics. The above example can be represented as the simple directed graph of Figure 7. The example, by virtue of being essentially a dataflow diagram, maps nicely onto a directed, acyclic graph. In general, though, more complex applications will require more complex graphs for their representations.

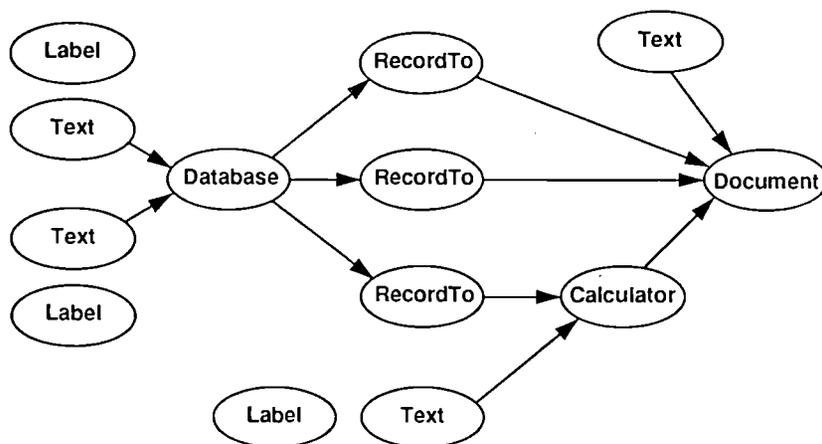


Fig. 7: Example script as a graph

Vista is implemented in C + +. The graph management of Vista is supplied by a set of class definitions and functions provided by the *Labyrinth* graphics editor [16]. Components and links displayed on the screen use the display capabilities of *Labyrinth* together with widgets created using the C + +/Motif binding software from the Univer-

sity of Lowell. The currently supported hardware platforms are the Sun SPARCstation and 386 machines running SCO UNIX.

The user of Vista is presented with the scripting concepts of component, port and link. At the implementation level these concepts are mapped into groups of cooperating objects. Each component has an *activity*, that is, the internal representation of its state and services, and a *visual presentation*, which includes its appearance and its user interface. The activity of a component is either implemented as a C++ object or, in the case of encapsulated scripts, as a graph. Limited support for activities implemented as Cool2 objects is also available. The visual presentation of a component is currently supported through Motif widgets. These two parts are kept separate to allow for the possibility of associating different presentations to the same component. The activity and presentation are connected and can communicate with each other via the graph structure representing the script. Each is wrapped inside a special *node* object that allows it to participate in the script.

Two components transfer information between each other when their ports are linked. Each port is represented in the graph by a special *port node* object and each link by a *link node* object. The port and link nodes are responsible for maintaining local and global constraints and to inform the user if invalid links are attempted. These constraints and linking rules are obtained from the scripting model.

Vista is intended to serve as an evolutionary prototype. The user interface of the tool is continuously changing as new functionality is added to Vista and as we get more insight into the possible ways of scripting applications. There is a wide variety of topics still left open in this first version of the tool which we plan to investigate. These topics include: a better way of adding new components, more support for components written in different programming languages, the general treatment of scripting models and generating code from a script.

6. Related work

There are presently a number of research prototypes and commercial products that illustrate the essence of scripting. Each of these tools, however, is based on a *single* scripting model for a particular component framework. We argue that, despite the success of some of these tools, the full potential for scripting applications is still largely untapped. Let us briefly consider three examples.

One such tool, Fabrik [15], is a visual programming environment that exploits bidirectional dataflow. The goal of Fabrik was to facilitate the *kit* approach to programming by taking advantage of emerging technologies such as iconic user interfaces. With kits, fixed rules govern the composition of kit components, thus restricting their possible reusability and the flexibility of application generation. Visual scripting, with scripting models, tries to factor out such restrictions to allow the creation of application templates that can be tailored to particular application domains.

HyperCard [14] is an authoring tool and information organizer based on the concept of *stacks* of information. HyperCard provides a fixed set of five components that can be configured interactively or through a high level programming/scripting language called Hypertalk. HyperCard is self-contained, no components can be added and there is no distinction between the tool environment and the application. In contrast, visual scripting supports the composition of components retrieved from continuously evolving

² Cool is the programming language developed within ITHACA - see section 7

component sets. The only restriction on new components is that they conform to the rules of a predefined scripting model.

Interface Builder [17] allows an application designer to define an application's interface graphically and to connect user interface objects to underlying application objects which have been programmed separately. Interface Builder was not intended to be an application generating tool, but it is gradually becoming more versatile in its functionality. Interface Builder, in contrast to visual scripting, does not support hierarchical design, that is, the encapsulation of scripts as components to be reused in future development.

Each of these tools has a very attractive direct manipulation user interface and attacks significant problems in the application development community but they are all limited to describing particular *instances* of applications whereas scripting in general can support the development of generic applications as scripts. Furthermore, by means of multiple scripting models, it is possible to support not only evolving component sets but to generalize scripting to different application domains each with its own standards for software composition.

7. A Scripting Environment

A visual scripting tool by itself is not sufficient to adequately support component-oriented software development. In addition, we need programming language tools to support the development of components, class management tools to support the organization, management, storage and retrieval of components, and tools to help in the task of matching requirements to available component frameworks.

Vista is being developed as part of the *ITHACA3* project [1]. The goal of *ITHACA*, which stands for "*Integrated Toolkit for Highly Advanced Computer Applications*," is to produce an advanced development environment for industrial applications based on object-oriented technology. This environment includes an object-oriented language called *Cool*, closely integrated with an object-oriented database, and tools to support the development of applications. A central component of the environment is the *software information base* (SIB) which stores and manages structured "descriptions" of software. The other tools interact primarily by exchanging and manipulating information stored in the SIB. They include: a *selection tool* for browsing and querying the SIB; *MaX*, a *monitoring debugger* for *Cool* classes; *RECAST*, a *requirements collection and specification tool*; and *Vista*. Standard tools and software components are provided by *application workbenches* for specific application domains supported by *ITHACA*, such as Public Administration and Office Systems.

In Figure 8 we can see the relationships between these tools. The purpose of *RECAST* [10] [11] is to provide a "guided" tour of the SIB [3]. It is this activity that aids the application developer in "negotiating" between the application requirements and the available software components. Without this negotiation, the chance of arriving at any reusable components as a result of an object-oriented analysis and design is significantly reduced. Since this is a critical activity, the SIB cannot merely be a repository of component frameworks. In addition, it must store and maintain application domain

3 *ITHACA* is a 5-year, 100 person-year/year Technology Integrated Project (#2705) in the Office & Business section of the European Community's Esprit II Programme. The partners are Siemens/Nixdorf (Berlin), Bull (Paris), Datamont (Milan), TAO - Tècnics en Automatització d'Oficines (Barcelona), FORTH - the Foundation of Research and Technology, Hellas (Iraklion) and CUI - the Centre Universitaire d'Informatique of the University of Geneva

models, requirements models, implementation "hints" etc.-in short, software descriptions that encode *experience*.

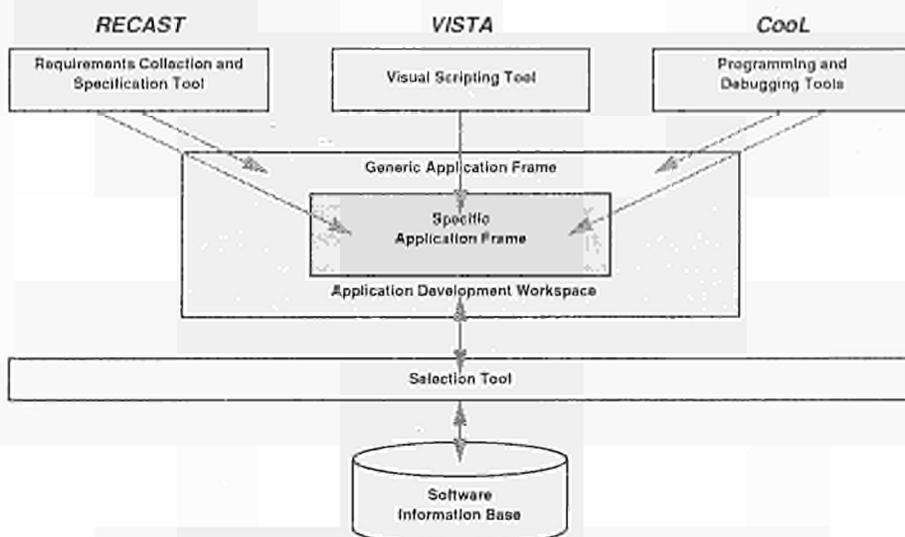


Fig. 8: The ITHACA Application Development Environment

The ITHACA term for one of these decorated component frameworks is a *Generic Application Frame* (GAF). The negotiation task is essentially one of retrieving and refining a GAF up to the point where a specific application can be largely scripted. If necessary, new components can be programmed at any point.

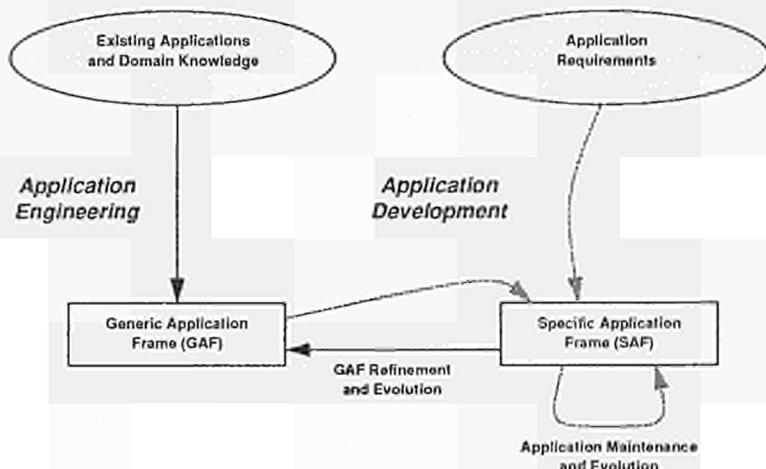


Fig. 9: The Evolutionary Software Life-cycle of Application Engineering

In the long term, we hope to make the job of the application developer as easy as possible. This requires, however, a substantial investment not only in developing the tools and environment, but in populating the SIB. The activity of developing scripting

models, reusable components and their associated GAFs is called *application engineering* (which is analogous to the development of product lines, whereas application development is analogous to sales). The job of the application engineer is much harder than the traditional one of a software engineer, as he or she must develop reusable software for unknown application with unknown requirements. The only plausible way to attack such a problem is to adopt an evolutionary software life-cycle in which previous and future experiences feed back into the iterative development of component frameworks and GAFs (Figure 9).

Within ITHACA, the job of the application engineer is being played by the developers of the application workbenches. A framework for office applications, called WooRKS, consists of a large collection of Cool classes that address various aspects of office work. A key component of WooRKS is COP, the run-time system for coordinating office procedures. (Steps in an office procedure consist of preconditions over a set of electronic documents and actions that transform these documents.) Office procedures can be defined using the Activity Definition Language (ADL) and compiled into Cool. As a practical application of scripting, Vista is being used as a graphical interface for designing office procedures. Scripts represent graphically the inputs, outputs, preconditions and actions of office procedures, and "execution" of an office procedure script causes the ADL specification to be automatically generated.

8. Concluding Remarks

Object-oriented technology addresses very well issues of encapsulation of state and behaviour, instantiation of objects through classes, and incremental modification and reuse of code and interface through inheritance and related mechanisms. But object-oriented programming alone does not necessarily encourage *component-oriented* development of applications—one must still *program* new classes to make use of existing ones. We have argued that *scripts* are the previously missing summand in the equation: *objects + scripts = applications*. It should be noted that end users need applications, not programs or objects or components.

Scripts serve to introduce and bind objects that, like character actors, know how to play a well-defined role. A *scripting model* defines and standardizes the possible binding relationships between components, each of which stands for some set of abstract services available at the level of the object-oriented language. A scripting model determines the standard interfaces for the reusable components of a component framework. Components that conform to the scripting model will be *plug-compatible*, and so can be scripted to produce application. Familiar examples include the stream-based scripting model of Unix shell scripts and the event-based scripting model of user interface toolkits.

A *visual scripting tool* provides a graphical, direct manipulation interface for the interactive construction and execution of scripts. We have implemented a prototype, called *Vista*, which supports scripting of both the behaviour and the user interface of applications. Vista supports the visualization and graphical linking of components selected from an extensible component set. Scripts can be interactively executed, saved, and encapsulated as components for use in other scripts. At present Vista provides a single, hard-wired dataflow-based scripting model. Vista has been designed, however, with the goal of eventually supporting multiple scripting models for a range of component frameworks. This implies the need for a standard notation for defining these models. More experience with various kinds of scripting models is required before such a notation can be formally defined.

Vista is being developed as part of *ITHACA*, an Esprit Technology Integration Project which seeks to produce a complete environment for application development based on object-oriented technology. Such an environment includes interactive programming and debugging tools, a Software Information Base for storing descriptions of software components and application domain knowledge, a selection tool for querying and navigating through the SIB, a requirements collection and specification tool to aid in the task of negotiating between application requirements and available component frameworks, and Vista, the visual scripting tool. *ITHACA* also provides, by means of a set of *application workbenches*, the means to test the tools and environment in a selected set of well-understood industrial application domains.

In addition to continuing work on Vista, we are pursuing a number of other directions closely related to visual scripting. Our first prototype of a scripting tool was *TEMPO*, a language for scripting temporal objects, in particular, animations [9]. Recent work on an object-oriented multimedia framework supports scripting [25]. We have also developed scripting interfaces for musical composition [6].

Another closely related topic is the specification of object-oriented programming languages that are better suited to the development of reusable software components. If one observes that object-oriented mechanisms supporting reuse, such as classes, inheritance, mixins, generics etc., all work in essentially the same way, that is by functional composition of various kinds of software components, then it is very natural to propose a *pattern* mechanism that generalizes this principle. In fact, scripts and components are both examples of "software patterns". A pattern language for active objects would be a natural foundation both for the development of reusable components and for scripting [20][21].

We shall close with a philosophical remark on the nature of scripting. Scripting is far from being a new idea-humans have used scripting to convey a general plan of action between loosely cooperating actors. Football coaches make up scripts to be executed by their players. Film directors outline scripts for their actors. Generals script battle plans for their troops. Programmers used to combine language statements to write programs. We suggest instead that they should script their objects to build applications.

Acknowledgements

Vista has been implemented by Vicki de Mey, Serge Renfer, Betty Junod and Marc Stadelmann. The precursor to Vista, VST, was implemented by Marc Stadelmann and Jan Vitek. We thank our *ITHACA* partners and associates for stimulating discussions on the nature of application engineering, GAFs, and scripting.

References

- [1] M. Ader, O.M. Nierstrasz, S. McMahon, G. Muller and A-K. Proffrock, "The *ITHACA* Technology: A Landscape for Object-Oriented Application Development," Proceedings, Esprit 1990 Conference, pp. 31-51, Kluwer Academic Publishers, Dordrecht, NL, 1990.
- [2] G. Anderson and P. Anderson, *The UNIX C-shell Field Guide*, Prentice-Hall, 1986.
- [3] P. Constantopoulos, M. Dorr, E. Pataki, E. Petra, G. Spanoudakis and Y. Vassiliou, "The Software Information Base - Selection Tool Integrated Prototype," *ITHACA* report FORTH.91.E2.#3, Foundation of Research and Technology - Hellas, Iraklion, Crete, Jan 12, 1991.
- [4] B.J. Cox, *Object Oriented Programming - An Evolutionary Approach*, Addison-Wesley, Reading, Mass., 1986.

- [5] B.J. Cox, "Planning the Software Industrial Revolution," IEEE Software, vol. 7, no. 6, pp. 25-33, Nov. 1990.
- [6] L. Dami, "Scripting Musical Components in Application Interfaces," in Object Management, ed. D.C. Tschritzis, pp. 357-366, Centre Universitaire d'Informatique, University of Geneva, July 1990.
- [7] V. de Mey, "Vista Implementation," ITHACA Report CUI.90.E.4.#1, Centre Universitaire d'Informatique, University of Geneva, Dec, 1990.
- [8] V. de Mey, "Vista User's Guide," ITHACA Report CUI.90.E.4.#2, Centre Universitaire d'Informatique, University of Geneva, Dec, 1990.
- [9] E. Fiume, D.C. Tschritzis and L. Dami, "A Temporal Scripting Language for Object-Oriented Animation," Proceedings of Eurographics 1987 (North-Holland), Elsevier Science Publishers, Amsterdam, 1987.
- [10] M.G. Fugini and B. Pernici, "RECAST: a tool for reusing requirements," in Proceedings CAiSE'90, LNCS 436, Springer Verlag, 1990.
- [11] M. G. Fugini, M. Guggino and B. Pernici, Reusing Requirements Through a Modeling and Composition Support Tool, Trondheim, May 1991, Accepted to CAiSE'91.
- [12] N. Gehani, "The Potential of Forms in Office Automation," IEEE Transactions on Communications, vol. Com-30, no. 1, pp. 120-125, Jan 1982.
- [13] S.J. Gibbs, D.C. Tschritzis, E. Casais, O.M. Nierstrasz and X. Pintado, "Class Management for Software Communities," Communications of the ACM, vol. 33, no. 9, pp. 90-103, Sept. 1990.
- [14] D. Goodman, The Complete Hypercard Handbook, Bantam Books, 1988.
- [15] D. Ingalls, "Fabrik: A Visual Programming Environment," Object-Oriented Programming Systems Languages and Applications (OOPSLA), Special Issue of SIGPLAN Notices, vol. 23, no. 11, pp. 176-190, Nov. 1988.
- [16] M. Katevenis, T. Sorilos, C. Georgis and P. Kalogerakis, "Laby User's Manual (version 2.10)," ITHACA report FORTH.90.E3.3.#7, Foundation of Research and Technology - Hellas, Iraklion, Crete, Dec 31, 1990.
- [17] NeXT Preliminary 1.0 System Reference Manual: Concepts, NeXT Inc., 1989.
- [18] O.M. Nierstrasz and D.C. Tschritzis, "Integrated Office Systems," in Object-Oriented Concepts, Databases and Applications, ed. W. Kim and F. Lochovsky, pp. 199-215, ACM Press and Addison-Wesley, 1989.
- [19] O.M. Nierstrasz, L. Dami, V. de Mey, M. Stadelmann, D.C. Tschritzis and J. Vitek, "Visual Scripting - Towards Interactive Construction of Object-Oriented Applications," in Object Management, ed. D.C. Tschritzis, pp. 315-331, Centre Universitaire d'Informatique, University of Geneva, July 1990.
- [20] O.M. Nierstrasz and M. Papathomas, "Viewing Objects as Patterns of Communicating Agents," ACM SIGPLAN Notices, Proceedings OOPSLA/ECOOP'90, vol. 25, no. 10, pp. 38-43, Oct 1990.
- [21] O. Nierstrasz, "The Next 700 Concurrent Object-Oriented Languages - Reflections on the Future of Object-Based Concurrency," in Object Composition, ed. D.C. Tschritzis, Centre Universitaire d'Informatique, University of Geneva, June 1991, pp. 165-187, Submitted for publication.
- [22] M. Shaw, "Prospects for an Engineering Discipline of Software," IEEE Software, vol. 7, no. 6, pp. 15-24, Nov. 1990.
- [23] M. Stadelmann, G. Kappel and J. Vitek, "VST: A Scripting Tool Based on the UNIX Shell," in Object Management, ed. D.C. Tschritzis, pp. 333-344, Centre Universitaire d'Informatique, University of Geneva, July 1990.
- [24] D.C. Tschritzis, "Form Management," CACM, vol. 25, no. 7, pp. 453-478, July 1982.

[25] D.C. Tschritzis, S.J. Gibbs and L. Dami, "Active Media," submitted for publication, Centre Universitaire d'Informatique, University of Geneva, Jan 1991.

[26] P. Wegner, "Capital-Intensive Software Technology," IEEE Software, vol. 1, no. 3, July 1984.

[27] R. Wirfs-Brock and R.E. Johnson, "Surveying Current Research in Object-Oriented Design," Communications of the ACM, vol. 33, no. 9, pp. 104-124, Sept. 1990.

Establishing the conditions for success in multimedia CD publishing

Willem Bulthuis - Philips

ABSTRACT

Interactive multimedia CD's will soon play an important role in our lives. A new publishing industry sector will emerge, in which Europe can play a leading role by exploiting both its technological and cultural strengths. In order to streamline the widespread uptake of the new medium and to optimise the changes for European business, it is essential to harmonise the requirements of all parties involved.

The OSMOSE project¹ aims at building bridges between technology providers and technology users. Its main charter is to create optimal conditions for developers and publishers of interactive multimedia titles. Therefore, OSMOSE is a strongly market-driven project. This paper outlines the starting points and approach of the project during its exploratory first year.

1. Introduction

Sponsored by the CEC under the ESPRIT programme, the OSMOSE project is an initiative of a group representing the Information Technology & Telecommunication industry, the Consumer Electronics industry, the Publishing industry, Software developers, and Research institutes.

The challenge of the project is to establish the optimal conditions for business success in selling CD titles and related tools. OSMOSE is achieving this by bringing together all parties involved in this new and exciting business: end-users, distributors, publishers, developers, copyright owners, network operators, tool builders, hardware manufacturers, standardisation bodies, and market researchers. It wants to bridge the gap between technology providers and those who want to exploit these technologies and be part of the new publishing age.

In its first year, the project has focused on finding out exactly what are the critical success factors in the interactive multimedia business. This has resulted in frameworks that help all parties to clarify their potential role in this business, and in clear specifications of what is expected from the technology providers (like development tools, application standards, etc.). In the second phase, the OSMOSE group is fighting to meet those expectations in a very short time frame.

2. The opportunities for multimedia CD publishing

Multimedia is a buzzword, but it also constitutes a real breakthrough in our information society. We will no longer be confined to sequential, text-based data, but we will

¹ The OSMOSE project (Open System for Multimedia Optical Storage Environments) was sponsored by the CEC as ESPRIT Exploratory Project 5656 from October 1990 to October 1991, with the following partners: Philips, Olivetti Systems & Networks, Bull, Maxwell Communications, Espasa Calpe, CAP SESA Telecom, Elektrosan, University of Athens, and subcontractor Industrial Market Research (IMR).

be confronted with a wealth of information reaching us via our natural senses: eyes, ears, and fingers. Moreover, this information will not be static and sequentially organised, but it will be dynamic, moving, and easily accessible.

The key to all this is not just *multimedia* (our current TV is already multimedia), but rather *interactivity*. Instead of watching passively from your couch, we will be in the drivers seat, control the information flow and act on it.

Multimedia is becoming affordable thanks to the incredible progress in digital and optical technology. The result is low cost equipment that is so powerful that we have problems imagining all potential applications. A whole new art and business is required to make optimal use of the emerging multimedia technology. It is unique that a new type of hi-tech products, like interactive CD's, will be earlier reaching the consumer market than the professional market. This requires different relations between technologists and marketeers, different business cooperation schemes, and different analysis of end-user requirements.

The use

Interactive multimedia information will come to us in all environments: at home, at school, at work, at the counter of shops and banks, and in public places like hospitals and post offices. We will be able to browse through information, learn from it, be entertained, and act on it by making decisions, placing orders, etc.

Interactive multimedia is a reality today. The necessary computer-based devices are available, whether in the shape of a low cost CD-I player or as a high-end multimedia workstation. Nevertheless, further integration of technologies is needed, as e.g. in the ESPRIT Multiworks project, to extend the capabilities and reduce costs.

The development

The key is the *development and publishing of applications*. In Europe, we have a tremendous potential in this respect. We have a very strong publishing history, an enormous cultural heritage and a wealth of sources for multimedia applications. The peripheral regions of Europe have unique chances to play an important role, and rural areas can benefit strongly by getting access to e.g. technology-based training and education which otherwise would only be available in the densely populated areas.

The distribution

There are basically three methods possible for distributing interactive multimedia material:

- **Distributing Compact Disks (CD-I or CD-ROM/XA)**
This is a very cheap distribution method, both for niche and mass markets; it is very convenient for the end-user, and a stable publishing standard.
- **Downloading and communication via networks (eg. ISDN)**
This method is complementary to CD-based distribution and is required for specific applications (up-to-date information, transactions, or realtime communication). However, it is not an alternative for distributing large amounts of multimedia material, since costs will be high and building the infrastructure will take a considerable time.
- **Broadcasting and local recording (eg. via TV satellites)**
This is potentially a cheap method for addressing mass markets, but it requires local digital storage facilities for later, interactive playback.

CD-based distribution will be widely used, and will follow the success of the audio CD because of its ease of use, low production costs, and worldwide standardisation. Moreover, it matches the traditional book publishing infrastructure quite well and is therefore very attractive for publishers. Of course we will also see combinations of the different distribution schemes. For example: the distribution of multimedia mail-order catalogues will be done on CD, while day-to-day price information as well as actual ordering might be handled via ISDN.

Printed material will not disappear in the near future. However, it will be enhanced and partially replaced by multimedia material. The publishing industry is eager to make this happen, but is dependant on e.g. a sufficient installed based of standardised multimedia players, sufficient distribution channels, and adequate CD title development facilities.

It is the objective of the OSMOSE project to analyse and establish the conditions for success in interactive multimedia CD publishing.

3. The critical success factors

The major goal of the first phase of the OSMOSE project is to explore the key success factors for the interactive multimedia CD business. Although the results of the OSMOSE User Requirements Survey are not yet included in this paper, we can already identify some of the key areas. Five factors seem to be critical for the success of interactive multimedia publishing:

- **adequate publishing and distribution scenarios**, defining the roles of all parties involved, from concept creator to customer, their collaboration schemes, the value added by each of them, copyrights, and money flows
- **availability of skilled CD-title developers**
- **availability of tools** to facilitate developing interactive multimedia titles
- **stability of publishing standards**
- **installed based of players**

Each of these factors is addressed by the OSMOSE project, either directly or in cooperation with other organisations or CEC programmes. A preliminary outline of the approach is presented in the next chapter, but we will first discuss each of the factors briefly.

3.1 Adequate publishing and distribution scenarios

Technological issues (standards etc.) are at present not the primary concern for publishers that want to become involved in multimedia. Their key question is how to make money in the changing publishing industry.

Each individual publisher is faced with the question what role(s) to play in the new multimedia game: copyright owner, risk taker, value adder, integrator, distributor, marketer, etc. New collaboration schemes have to be developed to deal with the very complex problems of copyrights and royalties. The investments involved in developing (a series of) multimedia titles often requires sharing the risks with other enterprises. Bespoke applications (produced on demand for a paying client) are simpler in this respect, but do not trigger mass markets.

Distribution and sales channels for interactive multimedia titles must be developed. The introduction of audio CD's was relatively simple in this respect, as it was just a new

technology for a well know product: music. Interactive multimedia is something completely new and requires new marketing and sales approaches.

The OSMOSE project addresses these issues by developing a framework for application & publishing scenarios, supporting all actors involved to define their future role in the multimedia business.

3.2 Skilled CD-title developers

A major hurdle in the development of good interactive multimedia titles is the shortage of creative and skilled developers. As mentioned before interactive multimedia is something completely new, implying that new paradigms must be developed. In addition to the key problem of how to design successful applications there is the problem of multidisciplinary. In the whole development cycle many different experts are involved: subject matter specialists, media experts, interactivity/art designers, programmers, application specialists (eg. for training, games, etc.). All these people have to work together and must be managed by the publisher.

OSMOSE is setting up cooperation with organisations that can stimulate the availability of skilled application developers (e.g. the Media Investment Club).

3.3 Development tools and environments

Many different experts are involved in the development of interactive multimedia titles. Therefore, there is a strong need for *project management* methods and tools, dedicated to CD title development. The sheer complexity of the material to be produced (large, multimedia, high level of interactivity) causes huge problems during the development process. Mechanisms and tools are required for *storing and managing all objects* that are produced during the development process (designs, specifications, multimedia objects, pieces of code). Furthermore, these object should be stored in such a way that they are *re-usable* for future title development projects, thus reducing the costs of title development. Finally, many different tools are used for the different parts of the whole development cycle, resulting in practical problems of *interchanging objects* of files between those tools.

These problems are very much like the situations we know from software engineering and mechanical engineering. In those areas much work has been done already on tool integration and re-usability (CASE, CAD/CAM), but in the area of interactive multimedia such approaches are just emerging.

One specific problem in the context of publishing is the required link between the production of printed matter and multimedia material. Many publishers want to exploit their resources for both types of publications, and thus want to have an integrated production facility.

The OSMOSE project considers it as one of its major tasks to develop a multimedia tool integration environment that also supports project management, objects management, and re-use of building blocks.

3.4 Stability of publishing standards

During the first discussions about setting up the OSMOSE project, standardisation was considered to be the key factor for success in the publishing business. After many discussions with publishers and developers it is now seen as *one of* the key factors.

Nevertheless it is a factor that requires very much attention, especially in the ESPRIT context.

Publishers can only be successful in the multimedia business if they can rely on a stable, worldwide standard with a large installed base. There is now one standard for interactive multimedia publishing that is stable and widespread enough to be accepted by publishers: the CD-I -- CD-ROM/XA Bridge standard, defined by Philips and Sony, and supported by many key players in the electronics industry. OSMOSE has been one of the promoters behind this bridge between the CD-I and CD-ROM/XA standards.

However, this is not yet sufficient to solve all application and distribution problems. A higher level *application standard* is required, that frees developers and publishers from all technical aspects of the different delivery platforms (CD-I, PC + CD-ROM-XA, Workstation + CD-ROM-XA, etc.). Moreover, this high-level application standard must cover the possibility to link a CD-based application to networks for updates, transactions, online communication, etc., as well as to other applications running on the local workstation (e.g. office automation tools).

*The establishment of such an application standard, called **OSMOSE-Talk**, is a key task for the project. However, this standard will not be developed from scratch, but largely as an integration of existing and emerging industry standards.*

3.5 Installed base of players

The issue of the installed base of interactive multimedia players is the traditional chicken-and-egg problem. Without sufficient applications people will not buy players, and without a sufficient installed player base publishers will not develop sufficient applications.

One approach to breaking this cycle is to establish a wide-ranging publishing standard, allowing one application to run on different player types. The CD-I -- CD-ROM/XA Bridge standard is an important step into this direction. Another important factor is the price of the player. Devices like the CD-I player will be affordable for home use, but more integration and cost reduction in multimedia workstations has to be realised.

The OSMOSE project addresses this issue by bringing together the end-users, developers/publishers and the technology providers, in order to stimulate the harmonisation of their specific interests at the earliest stage possible.

4. The OSMOSE approach

The OSMOSE project covers four major clusters of activities:

- User involvement and requirements surveys: **OSMOSE Interest Group**
- Prestructured Framework for Application/Publishing Scenarios: **PreFAPS**
- Application standardisation: **OSMOSE-Talk**
- Environment for Developing Interactive Titles: **EDIT**

For each of these the basic approach is to work together with other initiatives as much as possible, and to build on industry efforts where feasible. OSMOSE's strategy is to highlight crucial issues, to stimulate and harmonise ongoing work on those issues, and only perform those tasks that are not covered by others.

4.1 OSMOSE Interest Group

It is the objective of OSMOSE to involve as many players in the interactive multimedia business as possible, rather than acting as a small consortium. Therefore the OSMOSE Interest Group has been launched, which is open to any (potential) actor in this business.

The Interest Group functions as a bridge between multimedia technology developers and business people who want to exploit that technology in a profitable way. The group functions as information distributor concerning OSMOSE work and related issues, and it seeks sponsoring possibilities to stimulate the interactive multimedia title development. It is also a meeting point for people who want to set up a commercial collaboration.

The OSMOSE Interest Group is a group of people who want to contribute to and influence the evolution of the interactive multimedia business. It is planned to work closely together with related organisations and user groups. In the near future sub-groups will be created, with specific interests in e.g. using interactive titles, publishing, developing, copyright issues, authoring tools, standards, etc.

The Interest Group is the basis for the user requirements analysis performed by the OSMOSE project, in the form of workshops and surveys. These surveys are aimed at analysing the critical factors for successful uptake of interactive multimedia publishing. This analysis is the guiding element for all OSMOSE work.

It is at the heart of the project's philosophy to work on real needs, defined by real users of the technologies addressed by ESPRIT.

4.2 OSMOSE Framework for Application/Publishing Scenarios

In cooperation with publishing organisations and related institutes, OSMOSE is developing a *Prestructured Framework for Application/Publishing Scenarios (Pre-FAPS)*. This framework is already serving its purpose of identifying which roles companies could play in the interactive multimedia industry, how relations (e.g. copyrights, royalties) could be arranged, how distribution/retailing mechanisms could function, what application areas could be covered, etc. It functions as a guide for anybody who wants to get involved in interactive multimedia. Furthermore, the framework is used to analyse user requirements and to structure the OSMOSE work, to assure that OSMOSE reflects the organisation of the real business.

4.3 OSMOSE-Talk

OSMOSE-Talk is an integration and extension of established standards in the field of multimedia, at the level that is required for publishing titles that can run smoothly on different delivery platforms. There are already many multimedia standardisation activities around the world. OSMOSE does not intend to start another, competing with ongoing activities. However, it is felt that some issues that require special attention are not sufficiently addressed by other groups.

Firstly, most standardisation actions do not cover the full scope of interactive multimedia publishing. In principle this is a very good strategy, since it allows groups to focus on some limited part in order to make progress. On the other hand, it results in a lack of harmonisation between standards covering different areas. There is, for example, much work done on multimedia CD standardisation, and also on multimedia communication standards. But in a real application it is still a matter of improvising to

link all this together. Therefore, OSMOSE aims at *harmonising and integrating standardisation work* performed by existing groups, while focusing on the interests of the interactive multimedia publishing industry.

Secondly, much standardisation work is strongly technology-driven, mainly because technology is usually standardised before users know what to do with it. However, the increasing speed of technological innovations requires much faster uptake by real users. Therefore, it is essential to involve users much earlier in the innovation cycle and analyse their requirements already during the actual standardisation process. So OSMOSE aims at applying user requirements as a basis for standardisation work and focuses at high-level application standards (OSMOSE-Talk) rather than low-level technical standards. OSMOSE is in close contact with leading standardisation groups like ISO, ECMA, the CD standardisation companies (Philips & Sony), the Frankfurt Group (on CD-WO), Microsoft's Multimedia PC (MPC) consortium, other ESPRIT projects (notably Multiworks), etc.

In summary, OSMOSE-Talk will be based on user requirements, and built on top of emerging industry standards rather than competing with these.

4.4 OSMOSE Environment for Developing Interactive Titles

There is a strong need for more effective and efficient tool environments to support the development of interactive multimedia material. Multimedia editing tools are becoming widely available, but there are hardly any attempts to address the problems of large-scale, professional, team-based development of interactive multimedia titles.

OSMOSE aims at supporting the optimal exploitation of the many multimedia development tools available, by developing an open integration environment. Four topics are addressed:

- **project management** (coping with the multidisciplinary and complexity of the multimedia development process)
- **assets management** (managing the huge amounts of information objects)
- **re-usability of building blocks**
- **tool integration** (information integration & control integration)

In order to realise this, OSMOSE is developing standards for information exchange between tools (enabling both tool integration and re-usability), and for the description of development methods (rather than standardising the method itself). Furthermore, mechanisms are being developed to store, manage, and retrieve all information produced during the development process (through a Common Information Repository). Finally, tools are being developed to cope with the whole development process. To guarantee that third-party tools can be embedded easily in the environment, OSMOSE is seeking close cooperation with leading tool builders around the world.

The end-result will be an open tool environment, a set of tools and standards, as well as guidelines for encapsulating third-party tools.

5. Conclusions

Publishing interactive multimedia CDs will constitute a large part of the publishing industry in some years from now. In order for this business to have a smooth start it is essential to harmonise the interests of all actors at an early stage. Early cooperation between technology providers and technology users can reduce roadblocks and the risks for publishers.

OSMOSE has the mission to explore and establish the conditions for success in this interactive multimedia industry. The strongly market-driven approach of the project is rather new in the ESPRIT arena, but there is a clear need for such approaches. Already during its exploratory phase, OSMOSE has stimulated many business initiatives and started dialogues between technology-driven and market-driven people. It is essential now to turn this into a more structural part of the ESPRIT programme.

COMPUTER INTEGRATED MANUFACTURING AND ENGINEERING

ACTIVE VIBRATION CONTROL OF INDUSTRIAL ROBOTS FINAL RESULTS OF ESPRIT PROJECT SACODY

Jean-Luc FAILLOT
BERTIN et Cie
BP 3
78373 PLAISIR CEDEX
FRANCE

ABSTRACT.

Mechanical manipulators exhibit, when submitted to high speed and acceleration rates, vibrations which limit their performances and generate additional stresses on their structure. Up to now, robot designers have solved this problem by building stiff structures, yielding bulky, energy consuming, and expensive robots.

The basic idea of ESPRIT project 1561, called SACODY, was to propose an efficient alternative to this solution, relying on an intensive use of CAE methods such as Modal Analysis and Dynamic Modeling : To enable the design of high performance and cost effective robots, the robot designers must now turn to an advanced robot control to avoid structural vibrations.

SACODY, which ended in July 1991, succeeded in demonstrating on a spot welding robot an antivibration robot control running on an industrial hardware.

Beyond a completely new methodology of servo mechanisms control, enabling a tremendous improvement on the behaviour and performance of the industrial robots, the outcomes of SACODY are :

- A Commercial Software Package for Computer Aided Dynamic Analysis,
- a Software Package for the dynamic simulation of flexible multibody systems,
- a high performance Servo Level Hardware now incorporated in a commercialized Robot Control,
- prototypes of efficient Sensor systems for robot testing.

The paper recalls the project background and describes the project outcomes through their application to the project demonstrator.

1. SACODY Project background

Reducing cycle time of robots and manipulators without damaging their global accuracy is the traditional problem to be solved by robot manufacturers, who have to satisfy ever increasing specifications in terms of automated applications operational speeds.

Until now, roboticists have traditionally been relying mainly on the mechanical design of the robot to ensure its accuracy and avoid vibration phenomena. Should it lead to bulky and energy consuming systems, the robot structure is traditionally designed as stiff as possible.

Nevertheless, despite the use in the different robot components of very stiff and expensive materials, some applications, especially those involving large robots carrying heavy payloads, have their performances limited in order to avoid structural vibrations and their damaging effects on the system accuracy. This is for instance the case in fast pick and place maneuvers or, as it will be shown later, in spot welding applications,

where a non negligible time may be lost in order to allow for the tool vibrations be damped out before performing the desired task.

On the other hand, the low efficiency of current robots, in terms of nominal payload to structure weight ratio (typically 5 to 10 %), makes difficult to contemplate building ever stiffer mechanisms.

The idea that gave birth in 1987 to ESPRIT CIM project SACODY was to develop a new robot control strategy, relying on a better knowledge of robot structural dynamics, yielded by improved modeling and modal analysis techniques.

The work actually carried out within the framework of SACODY has enabled the definition of a new approach of robot control, able to ensure simultaneously fast motion of the robot and control of the vibrations of its structure.

SACODY ended in July 1991, by the successful implementation of the project developments for designing the active vibration control of an industrial demonstrator, made of an industrial robot and a prototype of high performance industrial robot controller.

SACODY, has been carried out by a consortium involving six european organisations namely:

- BERTIN et Cie (France, prime contractor),
- AEG AG (Germany),
- KUKA Roboter GmbH (Germany),
- The Katholieke Universiteit Leuven (Belgium),
- LMS International (Belgium),
- University College of Dublin (Ireland).

The sequel of this paper addresses in turns the different project outcomes by describing their application to the industrial demonstrator.

2. The project demonstrator

2.1 Demonstration application and demonstration robot

In order to better focus on the industrial and economical interest of the active vibration control, a spot welding application has been retained for the project demonstrator.

Spot welding remains the main robotized application, especially in the automotive industry where the spot welding robots represent 90% of all robots at work in car body assembly operations.

Because of their large range and the heavy tool they have to carry (up to 150 kg), these robots usually exhibit elasticities in their gears, yielding low frequency vibrations which damage the welding gun positioning accuracy, especially when maximum accelerations are sought.

The purpose of the demonstrator is to show that the application of the SACODY control algorithms enables to get rid of these vibrations and therefore improve the robot operational speed as well as the repeatability of the weld spot.

The demonstration robot system involves a KUKA IR160 industrial robot, designed for a 60 kg nominal payload and usually implemented in spot welding operations. This robot is shown on figure 1.



Figure 1 Demonstration robot : KUKA IR 160

2.2 The AEG High Performance Servo Level Robot Control

Although the problem of controlling flexible manipulators was an unsolved problem essentially dealt with by laboratory researchers, the SACODY consortium decided from the beginning to orientate the project towards a demonstration involving industrial hardware, in order to provide realistic results exploitable within a short leadtime at the industrial level.

This resolution has been implemented by AEG AG (Production Automation Group) who developed for SACODY a Robot Control with a High Performance Servo Level meeting the requirements formulated by the project team:

- High computation power for advanced control algorithms in order to enable the implementation of active vibration control for flexible mechanical structures,
- Flexibility with respect to the nature of sensors to be interfaced with the Robot Control (absolute encoders, incremental encoders, analog sensors (e.g. : tachogenerators, accelerometers,...),
- Reasonable flexibility with respect to the number of sensor measurements to be implemented for the control (conventional motor position and velocity measurements but also extra sensors measuring the vibrations of the robot),
- Flexibility with respect to the interface with servo drive systems in order to enable the integration of the control in the overall SACODY demonstration robot system, but

also to guarantee its applicability to flexible mechanisms implementing other drive technologies such as hydraulic actuators.

The work of AEG, consisting of hardware and software developments, has resulted in the delivery of an industrial Robot Control prototype featuring :

- A modular multiprocessor architecture,
- A new servo level environment able to incorporate advanced algorithms for anti-vibration position control and sensor signal processing,
- Enlarged Input/Output interface (the demonstration numerical control implemented 9 incremental encoders measurements and 3 tachogenerator analog measurements),

besides all the powerful control functions of a High Performance Robot Control namely :

- structured robot programming,
- off-line programming,
- JOINT, WORLD, TOOL, FRAME systems of coordinates,
- path planning and interpolation (Continuous Path Mode and Point To Point Mode),
- synchronous and asynchronous movement of up to 12 axes,
- override function for trajectories via a 3 dimensional joystick,
- macros for collision avoidance, deflection compensation and technology functions.

The following paragraphs address in turn the different design steps that have been carried out by the project partners to develop and implement the active vibration control on this demonstrator. They each focus on a specific design tool developed during the first part of the project and then applied to the specific case of the KUKA robot.

3. Computer Aided Testing and Dynamic Analysis of non-rigid mechanism

3.1 LMS CADA-X software

The experimental assessment of the dynamic robot behaviour is required to provide the necessary information for the optimization of the robot design as well as for the design of a controller which can take these dynamic characteristics into account.

In the framework of SACODY, a frequency domain identification technique deriving directly the parameters of a reduced state space model of a flexible robot has especially been developed (1).

This identification technique features the following characteristics :

- The state transition, input and output matrices of the state equation are readily available,
- Numerically reliable tools are used : least squares solution and eigenvalue decomposition techniques,
- The number of unknowns is limited, using only a second order linear model with matrix coefficients of dimensions equal to the number of modes. This model is valid for a reduced state. Since the output matrix is obtained from a real symmetric matrix, the usual singular value analysis is replaced by an eigenvalue decomposition, drastically reducing computer requirements and load as compared to traditional methods (e.g. ERA),
- A very important problem, namely the dimension of this state space model, is eliminated by the data measurement matrix,
- All measurement data can be analysed simultaneously, reducing the measurement noise, and yielding a consistent model for the system's response.

The integration of this frequency domain direct parameter identification (FDPI) technique into the LMS CADA-X software package for Computer Aided Testing (CAT) has been completed. This package has been set up for general dynamic signal acquisition and analysis, and contains the necessary hard- and software to perform for example modal analysis tests of mechanical and mechanical/servo systems. This CAT system also provides general software libraries for modern linear algebra, least squares problem solving, eigenvalue and -vector decomposition, singular value analysis, system theory, etc...

The integration of the identification algorithm in the LMS CADA-X system for computer aided dynamic analysis (CADA) and computer aided testing (CAT) makes communication with other software modules straightforward. The measurement data can be retrieved directly from the relational CADA-X project data base, making all necessary information available: used test equipment and circumstances, transducer sensitivity and calibration values, the needed frequency response functions, etc... The obtained modal parameters can be stored to the analysis tables of the same data base, and interpreted via a graphical animation module to display the deformation patterns (mode shapes). Other interesting analysis information (such as the singular values, indicating the state space dimension) can also be stored to this data base for later use.

3.2 Computer Aided Dynamic Analysis of the demonstrator

A dynamic analysis has been carried out on the demonstration robot in order to determine the dynamic model for this robot in the low frequency region.

The characterization was mainly achieved in two different ways. First, experimental modal analysis was used to describe the dynamic behaviour in terms of natural frequencies, damping values and modeshapes. These were determined out of frequency response functions of acceleration over a force being externally generated with a vibration exciter. Next, frequency response functions with internal excitation were measured. In this case, the excitation was generated by each of the first three motors separately.

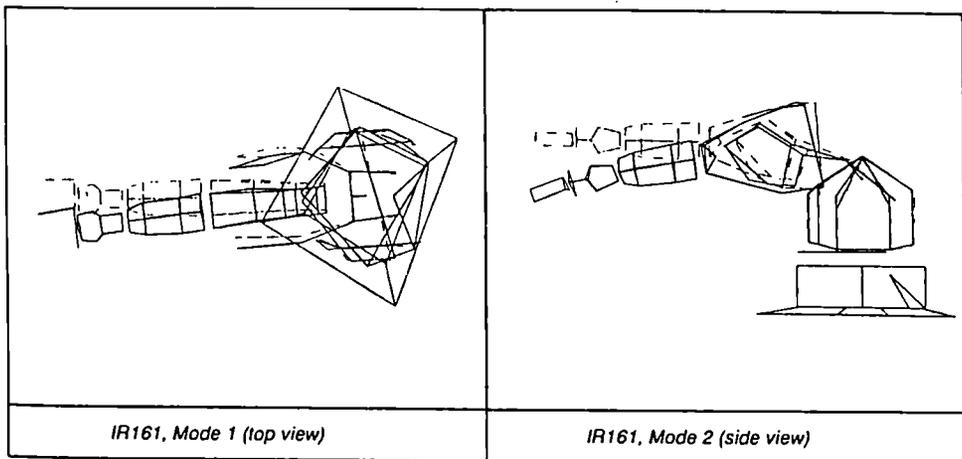


Figure 2 Two first vibration modes of the demonstrator robot

As the dynamic properties vary with the configuration of the robot, the analysis was carried out for different position configurations. The modal analysis was carried out in three configurations. The frequency response functions with respect to the motor current were measured in ten different configurations.

Four robot modes and one foundation mode were found in the frequency region of interest. The first two robots modes are one horizontal and one vertical rotation due to flexibilities lumped in the motor gearings as shown on figure 2. The two following modes are a horizontal and vertical bending with global structure deformation.

The study has therefore enabled to focus on the presence of elasticities in the robot which can be easily explained by the difficulty for the mechanical design to stiffen the robot gears in order to suppress transient oscillations and static deflections, especially in this case where a heavy load (45 kg) has to be carried with a 2.5 meter range.

An active vibration control has therefore be defined in order to prevent the oscillations induced by these elasticities and therefore lessen the time needed to position the robot tool with a prespecified accuracy.

4. Numerical modeling of flexible multibody systems

4.1 BERTIN ADAMEUS software

Computer dynamic simulations are more and more used for the design of complex mechanisms in order to validate design trade-offs without building expensive test prototypes that are moreover less able than simulation to reproduce all the features of the target system (e.g. non gravity tests for space applications).

The complexity of the couplings between the dynamic motion of a mechanism and the vibrations of its structure makes mandatory the availability of a dedicated computer code in order to understand the interaction of both phenomena and, which is the final purpose of SACODY, control it.

During the past decade, a lot of development has been devoted to the dynamic simulation of flexible multibody systems with application to spacecraft, satellites and manipulators. Most of these developments have been carried out in the United States, in the framework of large space research programs. Among the resulting software, only a few packages allow for the design of control for flexible multibody systems and its verification by means of closed loop simulations.

The objective of SACODY, in the field of computer modeling, was the adaptation of a general purpose computer code developed by BERTIN, called ADAMEUS, to the particular case of robots or manipulators with possible flexible links or joints. The resulting version enables the modelling and closed-loop simulation (i.e. simulation with a feedback control) of any mechanical polyarticulated chain, including actuators and sensors, as well as the derivation of linearised models for the purpose of modal analysis and control design.

ADAMEUS involves an adapted method for obtaining automatically dynamical equations of flexible mechanical systems. It enables a general approach of the problem of flexible systems dynamics for complex bodies behaviours and system topologies (open or closed chains with possible constraints).

The equations are built by use of the principle of virtual work. Rigid body kinematics uses relative coordinates, and a modal synthesis method, based upon the results of a Finite Element Analysis of the system's flexible components, is implemented to solve the problem of flexibility.

Thanks to new developments carried out during SACODY, the models yielded by ADAMEUS can be used not only for providing open-loop and closed-loop simulations, but also to compute:

- the kinematic inversion yielding the trajectory of joint coordinates while taking static flexibility into account,
- the predictive feedforward resulting from dynamic inversion along the desired trajectory,
- linearized models about prescribed trajectory points, to be handled by control design software to obtain the feedback gains needed by the servo control stage.

All these items are loaded in a control data file, and used in closed-loop simulations for computing the control orders to be applied to the actuators.

4.2 Model and Simulation of the Robot

A base model (figure 3) has been established by describing all the mechanical elements (links, speed reduction stages, motors,...) of which the robot is composed and their interconnection. The model features globally 45 connection nodes, 20 gearlike links and 16 rigid links or constraints.

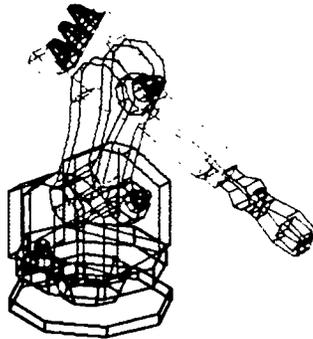


Figure 3 ADAMEUS model of the demonstration robot

In accordance with the results of the identification tests described above, two gearing shafts have been defined as flexible bodies in order to model the flexibilities lying in the joints of the first two axes. By proper choice of the shaft stiffnesses introduced in the model, it has been possible to reproduce the measured modal frequencies and their evolution with the robot configuration with an accuracy of 3%.

After optimization of the number of variables and the automatic elimination of the non-working degrees of freedom, 8 variables are kept by ADAMEUS, out of the 47 declared in input, namely the six motor rotations and the two shaft torsions.

5. Control design and implementation

5.1 Active vibration control specification

The first two years of SACODY have been devoted to the development of new control algorithms able to compensate for the deflections and vibrations of manipulators (2), (3). These developments have been led in close collaboration between BERTIN and K.U.Leuven. The antivibration algorithms have been applied and validated in the laboratories of the K.U.Leuven during the first part of the project on a series of

prototypes representative of flexible servomechanisms. Details on these experiments can be found in (4).

The final step of SACODY has been the application of these new methods to the project demonstrator. The specifications of the control to be designed were :

- To control and damp out the vibrations of the two first robot modes as identified by the LMS modal analysis campaign,
- to ensure the system's stability despite the existence of high frequency ignored modes,
- to track simple trajectory such as classical trapezoidal velocity profile,
- to maintain as far as possible the robot positioning time on the different axis, i.e. the time taken by the robot to reach for the first time the prescribed location.

The measurements being implemented in the control are the motor positions and velocities as well as measurements of the actual arm positions, yielded by additional sensors located after the joint flexibilities.

5.2 Active vibration control implementation

The final step of SACODY has been the design and the implementation of the active vibration control on the demonstrator described in chapter 2.

The control design has been led in two steps :

First, a preliminary real control time implementation has been carried out by the team of K.U.Leuven who applied the SACODY algorithms to the demonstration robot through a powerful laboratory multiprocessor by K.U.Leuven. This preliminary implementation enabled the test of various advanced control schemes, addressing servo control but also trajectory generation, but also the test in a user-friendly development environment of the control schemes likely to be implemented on the industrial demonstrator, which characteristics were specified by BERTIN following a trade-off between scientific feasibility and industrial applicability. Details on the K.U.Leuven work can be found in (5).

Starting from the results reached by the K.U.Leuven, the final implementation of the antivibration servo control in the AEG demonstrator has been carried out by BERTIN, with the assistance of AEG. The final structure of the control has been specified after simulations studies involving the model established with ADAMEUS and taking the AEG robot controller specificities into account.

It resulted in an active vibration control of the three main axis entirely implemented at the controller's servo level and treating conventional trajectory commands released by the controller system level (i.e. level in charge of kinematic transformations, interpolations, dialogue with operator), on which no modification has been brought.

The level of performance improvement reached is described next chapter.

6. Robot performance testing

6.1 Performance criteria for robot

SACODY has ended by the evaluation of the performance improvements achieved on the demonstrator by the new antivibration control algorithm. To this aim, a complete test procedure has been established by KUKA (6), following the recommendations of the ANSI/RIA R15.05 and ISO/DIS 9283 International Standards. More precisely, the following measurements were implemented :

6.1.1. Minimum Positioning Time

This measurement provides the quantification of the time taken by the robot to travel from its starting pose to its first arrival to the target pose.

6.1.2. Pose Stabilization Time and Pose Overshoot

This measurement, coupled with that of the Minimum Positioning Time, gives a complete characterisation of the robot positioning behaviour. The Pose Stabilization Time is the elapsed time between the first arrival of the robot to the target pose and its actual stop. It characterizes explicitly the vibrating behaviour of the robot.

The location of the measured poses as well as the definition of the measured quantities are shown on figure 4 extracted from ISO/DIS 9283.

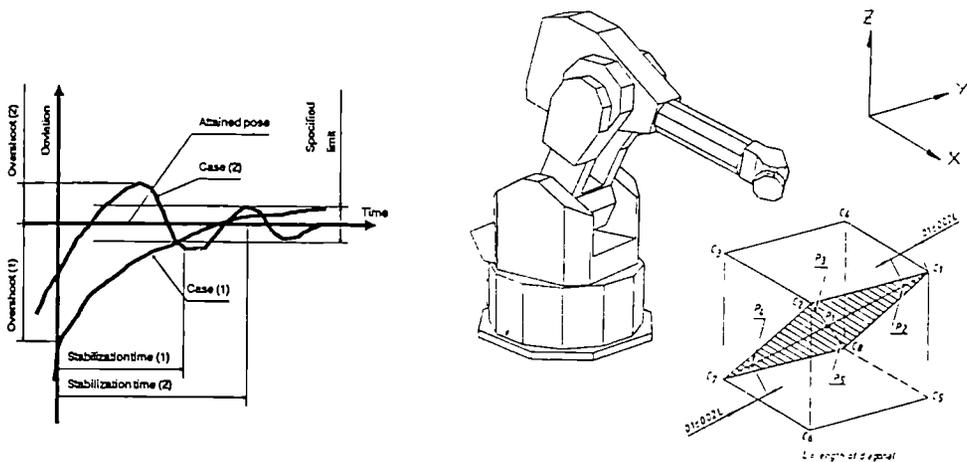


Figure 4 ISO definition of the pose stabilization time and pose overshoot

6.1.3. Cornering Deviations

Cornering Overshoot and Cornering Round-Off are the deviations measured between a commanded square continuous path and the actual path followed by the robot tool. These quantities are specified on figure 5 along with the test path implemented.

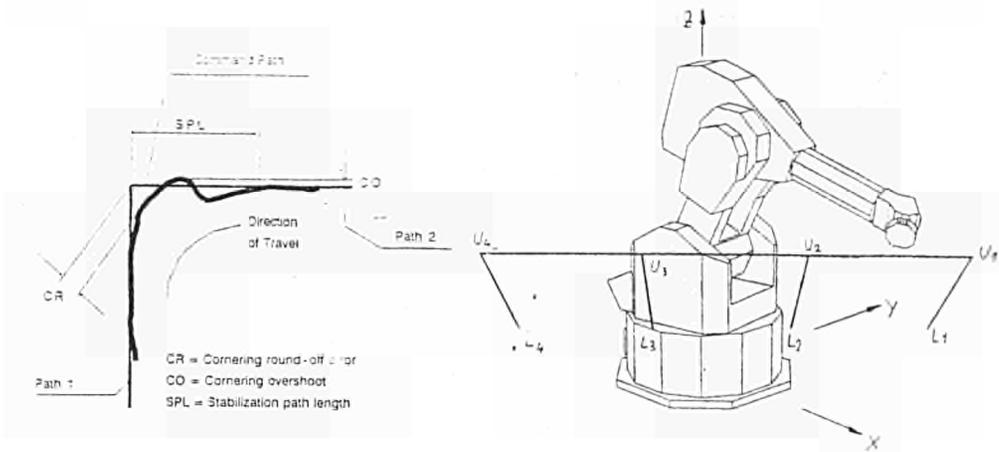


Figure 5 ANSI/RIA definition of the cornering deviations

6.2.K.U.Leuven sensor systems for robot performance testing

6.2.1. RODYM

One of the important aspects of the SACODY project has addressed the development of sensor systems for robot performance testing.

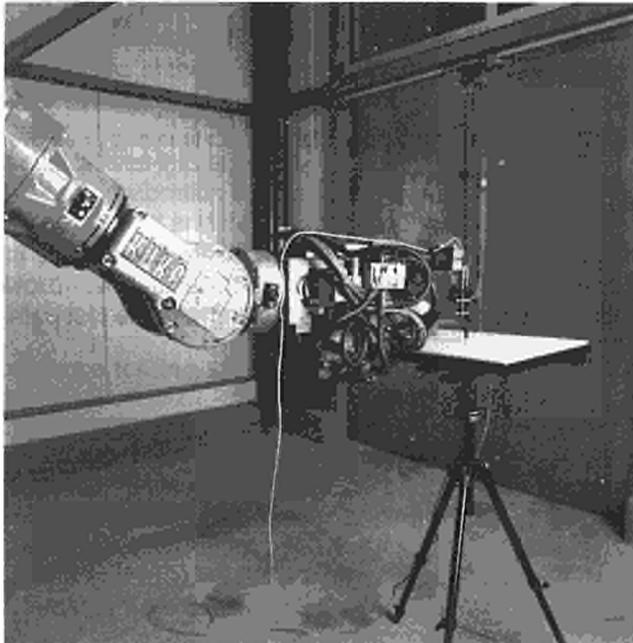


Figure 6 RODYM digitizing tablet

A first measurement system for robot dynamic performance analysis, called RODYM, has been developed and constructed in the laboratories of K.U.LEUVEN. It has been used for assessing the gain in performances obtained on the demonstration robot, by application of the test procedure overviewed in 6.1.

The RODYM system consists of a digitizing tablet, interfaced to a personal computer by an IEEE-bus, and a software package for data acquisition and analysis. X, Y coordinates of a measurement probe relative to the surface of the tablet are acquired by a personal computer. To perform tests, the probe is fixed at the end point of a robot. The tablet is positioned in such a way that the programmed robot movements are in the measurement plan of the board (figure 6).

Different Figures of Merit, ranging from overshoot, settling time to static position accuracy and path accuracy are calculated by a menu-driven software package. These figures of merit describe the performance characteristics of the tested robot. The results of a series of tests are reported in a performance data table.

The surface of the digitizer tablet can range up to 1 800 x 1 200. Two modes of acquisition are provided : point to point mode and stream mode. In the first mode a point is acquired continuously at the maximum sampling frequency. This mode is used for dynamic measurements.

The position of the probe can be acquired up to a distance of 22 mm orthogonal to the surface. When the probe is leaving this region, a signal is sent to the personal computer. This feature is used to synchronise RODYM and the robot under test. RODYM also detects if the robot has reached a commanded point, within a repeatability zone.

The accuracy of the sensor is improved with a factor of four by a calibration algorithm, up to 0.05 mm, which is in most cases at least an order of magnitude better than the robot. A high bandwidth (200 Hz), portability, a large measurement area (1 800 x 1 200 mm) and a user friendly menu driven software package, make this system very attractive for robotic application.

6.2.2. LASER-TREK

LASER-TREK is a 3D laser tracking system allowing the measurement of the linear coordinates of a point moving in three-dimensional space. To achieve this, a retro-reflector (corner cube or cat's eye) is connected to the object to be tracked (e.g. a robot end effector). The optical center of the retro-reflector forms the moving target. The measurements are contactless, the only connection between the measuring device and the measuring point being two laser beams, each originating from a deflection unit and pointing towards the optical center of the moving target. The deflection units, positioned around the moving target track the target by keeping the laser beams pointing towards it. To this aim, the deflection mirrors are mounted on galvanomotors which positions are controlled in order to maintain to zero the tracking error between the incident and the reflected beam positions, measured by a Position Sensitive Device. The position of the target is determined by triangulation using the two angular position of each of the two deflection mirrors. The measuring principle is shown in figure 7.

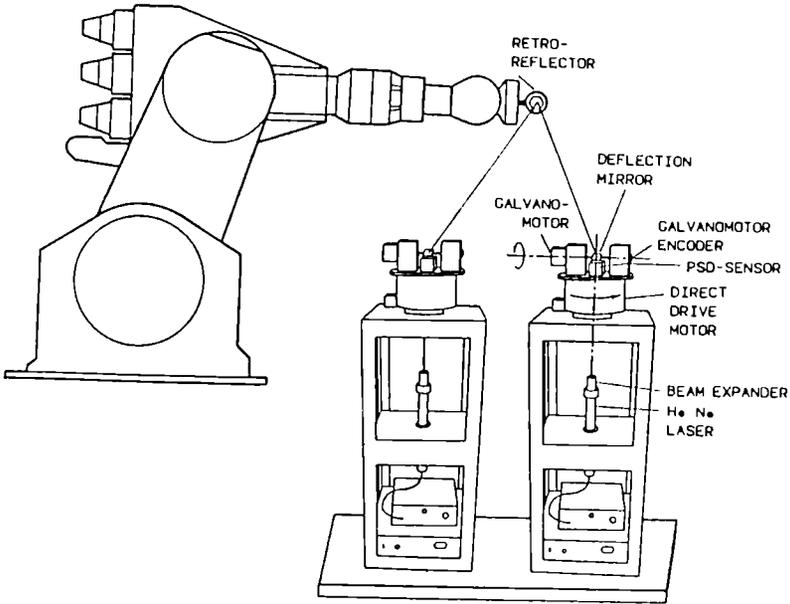


Figure 7 Measuring principle of LASER-TREK

The performance of LASER-TREK has been extensively tested using a coordinate measuring machine (CMM). This CMM generates accurate reference position data. Tests have been executed in a measuring space of 300 x 300 x 250 mm, laid-out according to figure 8.

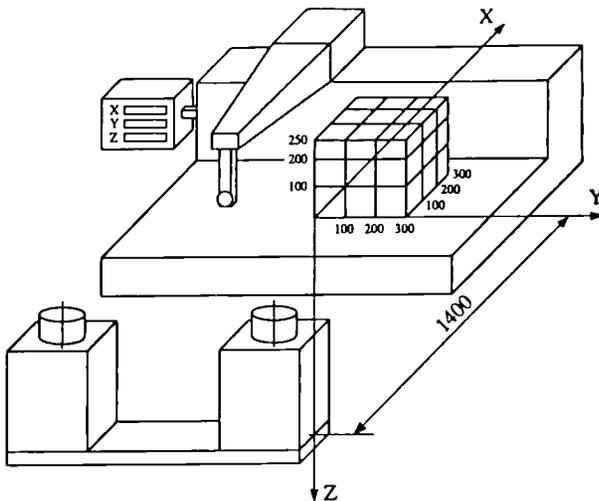


Figure 8 Calibration setup and calibration space

These performances are as follows :

- The distance measurement accuracy is :
 - along X-axis : 0.25 - 0.28 %,
 - along Y-axis : 0.06 - 0.14 %,
 - along Z-axis : 0.03 - 0.06 %.
- The resolution is :
 - along X-axis : ± 0.04 mm,
 - along Y- and Z-axis : ± 0.01 mm.
- The measurement noise is :
 - along X-axis : ± 0.04 mm,
 - along Y- and Z-axis : ± 0.01 mm.

The discrepancy between the accuracies along the X-axis with respect to those along Y- and Z-axis are completely due to the condition of the measurement setup. Due to the limited opening angle of the corner cube retro-reflector, the two deflection units have to be positioned relatively close to each other. The use of a cat's eye would enhance the measurement conditions considerably.

It has to be said that above remarkable results are **absolute** measurements. It is clear that the accuracy is much higher when comparative measurements are performed.

6.3 Robot performance improvement

The test procedure overviewed in 6.1 has been applied to the demonstrator controlled first by a conventional control scheme, then by the antivibration servo control.

The robot was submitted in both cases to the same trajectories, corresponding to the maximum accelerations achievable by the robot.

Although optimised, a conventional control approach does not manage in suppressing overshoots and vibrations due to the robot flexibilities. The overshoots observed range from 5 to 10 mm and the stabilization time is about 0.5 second.

For the sake of concision, we summarize the improvements achieved by the SACODY antivibration control.

For a given Point-to-Point motion :

- the Minimum Positioning Time, which does not include stabilisation time is longer of about 70 msec, but:
- 7 mm Overshoots are almost suppressed, and hence, an average 500 msec stabilisation time is saved.
- cornering deviations are smoother due to suppression of cornering overshoots.

For commanded motions lasting from 0.3 second to 2 second, a 430 millisecond time saving is achieved. Once turned into percentages, the positioning time saving reached by the antivibration control for a given repeatability ranges from 20 % (large motions) to 60 % (short motions).

Figures 9 and 10 overview qualitatively the improvements brought on the robot behaviour by the SACODY antivibration control.

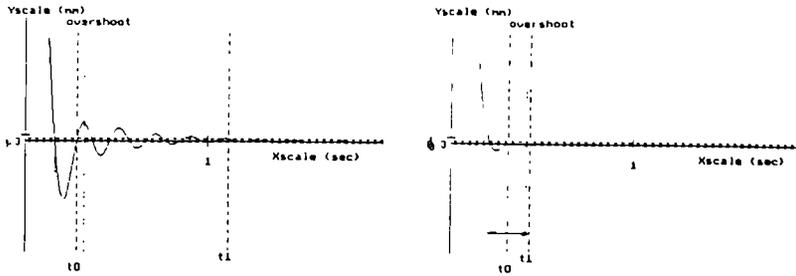


Figure 9 Suppression of pose overshoot by active vibration control

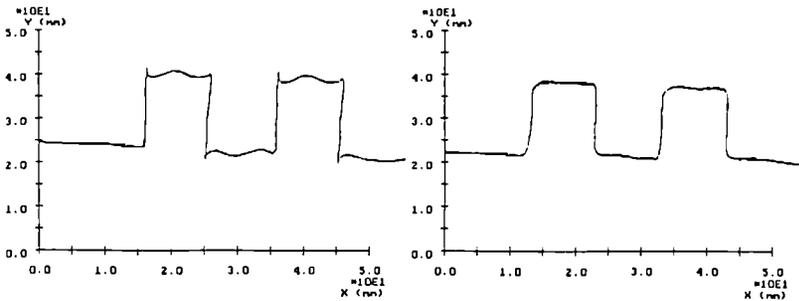


Figure 10 Smoothing of continuous path following by active vibration control

7. Conclusion

This paper has described the final results of the ESPRIT project SACODY, obtained on a demonstration robot system involving industrial hardware.

Besides the feasibility of active vibration control, the SACODY control approach has shown significant improvements in terms of robot system behaviour and performance.

The main advantages of the new control are :

- an important gain in robot cycle time,
- a robot positioning without overshooting and vibrations,
- a reduction of dynamic stresses on the robot structure.

This control, which runs on state of the art industrial controller boards offers a software alternative to the structure stiffening traditionally used to avoid robot vibrations.

The fields of application of the antivibration control demonstrated by SACODY is very wide, and covers all systems such as industrial robots, large manipulators, gantries, cranes, telescopic structure that suffer from lack of structural rigidity.

SACODY project has also shown how new control techniques relying on Computer Aided Testing and Dynamic Modeling can improve the productivity of current industrial Robot Systems.

Beyond a completely new methodology of servo mechanisms control, enabling a tremendous improvement on the behaviour and performance of the industrial robot, the outcomes of SACODY are :

- A Software Package for Computer Aided Dynamic Analysis commercialized by LMS,

- an upgraded Software Package for the dynamic simulation of flexible multibody systems to be commercialized by BERTIN,
- a high Performance Servo Level Hardware now incorporated in a marketed AEG Robot Control system,
- prototypes of sensor systems for robot testing, to be made available for commercial exploitation by K.U.Leuven.

The SACODY breakthrough in robot control now paves the way for the development an integrated design of robot systems and servo mechanisms, where active vibration control capabilities should be taken into account for building lighter, cheaper and ever more efficient robots.

8. Acknowledgements

The work described in this paper has been achieved thanks to the collaboration of six European Partners, within the framework of ESPRIT Project 1561.

Such a collaboration would not have been possible without the help of the Commission of the European Communities, which we would like to acknowledge here.

Such results would not have been achieved either without the trust and the personal involvement of Dr Rainer ZIMMERMANN, who acted as Project Officer on behalf of the Commission. These few lines are also intended to acknowledge his constructive attitude and the dialogue he has maintained with the consortium all over the project.

9. References

- (1) Lembregts, F., Leuridan, J., Van Brussel, H., *Frequency domain direct parameter identification for modal analysis : State space formulation*, Mechanical systems and signal processing (1990) 4(1), pp 65-75.
- (2) De Schutter, J., Van Brussel, H., Adams, M., Froment, A., Faillot, J-L., *Control of flexible robots using generalized non linear decoupling*, IFAC Symposium on robot control SYROCO, Karlsruhe, October 1988.
- (3) Froment, A., Faillot, J-L., Gallay, G., Gerbeaux, F-X., *Modélisation et commande des structures flexibles polyarticulées*, Colloque SMAI "l'Automatique pour l'aéronautique et l'espace", Paris, Mars 1989.
- (4) Swevers, J., Adams, M., De Schutter, J., Van Brussel, H., Thielemans, H. *Limitations of linear identification and control techniques for flexible robots with non linear joint friction*. International symposium on Experimental Robotics, Montreal, Canada, June 19-21, 1989.
- (5) Swevers, J., Torfs, D., De Schutter, J., Van Brussel, H., *Comparison of control algorithms for flexible robots implemented on a KUKA IR 161/60 industrial Robot*. Proceeding of the fifth International Conference on Advanced Robotics, Italy, 19-22 June 1991.
- (6) Gerung, M., *Accuracy Measurement on a KUKA IR 160/60*. Series of SACODY Technical Reports, KUKA Schweissanlagen + Roboter GmbH 1990 - 1991.
- (7) Wunderlich, H., *Development of a Controller for a High Performance Servo-Level*, Presentation at the Final Review Meeting of ESPRIT Project 1561, Augsburg, Germany, 4 July 1991. AEG Aktiengesellschaft, Production Automation, Stuttgart Böblingen 1991.

NEUTRAL PRODUCT DEFINITION DATABASE FOR LARGE MULTIFUNCTIONAL SYSTEMS - NEUTRABAS -

F. FERNANDEZ-GONZALEZ

Universidad Politecnica de Madrid
Departemento de Arquitectura y Construccion Navales
Avenida Arco de la Victoria s/n
E - 28040 MADRID
Tel. : (34) 1 534 3709

M.G LEHNE

Bremer Institut für Betriebstechnik und
Angewandte Arbeitswissenschaft an der
Universität Bremen
Klagenfurter Straße/betriebshp
Postfach 33 04 40
D - 2800 BREMEN 33
Tel. : (49) 421 220 0955

R. VOPEL

Technische Universität Berlin
Sekt.SGIO
SALZUFER 17/19
Geb. 12
D - 1000 BERLIN 10
Tel. : (49) 30 314 24048

ABSTRACT

The paper describes the developments and results for a neutral product definition database for large multifunctional systems within the ESPRIT Project 2010 (NEUTRABAS). The database development is based on the product modeling methodology of the evolving ISO Standard STEP and its information modeling language EXPRESS. The main application areas envisioned for the NEUTRABAS Database are ships and other maritime products as representative prototypes of large and functionally very diverse systems. The emphasis is placed on methods for describing the structural systems, the spatial arrangements, and the multitudinous outfitting systems in this type of product.

The neutral database formats are intended to serve for purposes of file data exchange between heterogeneous CAD or CIM systems as well as for data sharing by means of a dynamic access neutral database. The results of the project include concept designs and specifications for the product information models and for the database components.

1. INTRODUCTION

The ESPRIT Project 2010, NEUTRABAS, is aimed at the development of a neutral product definition database for large, multifunctional systems. Many more complex engineering products serve several different functions simultaneously or sequentially and so do their constituent parts and subsystems. The multifunctional aspect creates special requirements for the methodology of product information modeling and database design. In NEUTRABAS, ships and other maritime products are regarded as a representative example and ideal test case for large, multifunctional systems and their product models. In addition, these products typically are of a one-off-production type.

The benefits of providing standardized product databases supporting a neutral data exchange and archiving capability are well known and apply also to the maritime industry.

They include:

- improved communication within enterprises and with outside partners by linking heterogeneous systems,
- integration throughout the CIM cycle, based on a single, coherent product model,
- distribution, decentralization, and coordination of activities within concurrent engineering environments,
- collaboration with other industries via open software bridges.

To realize this potential the design of the NEUTRABAS database is required to support the following properties :

- neutral, i.e, based on a standardized information modeling methodology, essentially the evolving ISO standard STEP and its modeling language EXPRESS (ISO CD 10303-11), as well as a standardized maritime product information model, which is being developed by NEUTRABAS,
- complete, i.e, comprising all entities and attributes subservient to all currently relevant product functions at all pertinent stages of the life cycle,
- structured, i.e; flexible in access, also from several different functional viewpoints,
- open, i.e, suitable for decomposition and distribution in open systems environments.

To develop such a database the following basic approach is taken by the NEUTRABAS project :

- information analysis of complex maritime products, in particular ships, using STEP based methodology,
- development of a ship product information model with a major emphasis on the structural configuration, spatial arrangements, and outfitting systems,
- design of a database conceptual schema for this product model with appropriate data management capabilities,
- specification of components for this database, with a main orientation toward relation database technology (SQL, ORACLE), but with object-oriented options in mind for future extensions,
- verification by pilot implementations, primarily for ship structures, potentially for a few outfitting systems, too,
- contributions toward STEP Application Protocols for ships and other multifunctional products.

Current results of these developments are reported in the following sections.

The NEUTRABAS Project has benefited much its dialogue and cooperation with the NIDDESC project group in the U.S NIDDESC (Navy Industry Digital Data Exchange Standards Committee) is a project with similar objectives in shipbuilding product modeling, which started some time before NEUTRABAS and has contributed valuable working documents to ISO STEP on the subjects of Ship Structural Systems and Distribution Systems Reference Models (GERA 90), (Mart 89). NEUTRABAS has been able to build on this work and to extend the scope and functionality of the product model.

2. MODELING LARGE, MULTIFUNCTIONAL SYSTEMS

The size and complexity of large, multifunctional products such as ships necessitates a special, structured and well coordinated approach to information modeling. This is reflected in the modeling method and in the choice of modeling tools.

For NEUTRABAS, it is the objective to capture in its database the substance of the product definition data as well as a data description of its functional capabilities and behaviour. A complete model, which includes all aspects of product functionality, would require an information structure for the product and for the functional processes in which it is involved. Currently, however, the main emphasis in international standards for product modeling such as ISO STEP, Version 1.0 is still placed on product definition data ; a new task for modeling Product Functionality is only just now being defined (in P17 of WG3/WG4/TC184). The available modeling tools in STEP, i.e. the information modeling language EXPRESS, focus on defining a product model as a structured data object in terms of all relevant product attributes. NEUTRABAS, too, in choosing EXPRESS as its modeling language, must concentrate on describing all product properties including product functionality by means of information elements or attribute sets, which are sufficient to evaluate the functional capabilities of the product. Process modeling is not part of the current scope of NEUTRABAS.

A product function is a role that a product performs in any given context or from any assumed viewpoint. Primary functions are those for which the product is designed or destined (purpose roles). Secondary functions are related to any other kind of functional behaviour of the product in some context or environment (behaviour roles). The distinction between purpose and behaviour is not always strict. The definition of functions depends on the viewpoint of product evaluation. Similarly, there is no strict limit to the number of functions a product may perform.

Consequently, it is necessary to define which functions will be included in a product model. NEUTRABAS has chosen a finite, though extensible set of functions to be taken into consideration. The model thus will be only as complete in its scope of product functionality as it aspires to be for the moment. The selection of function is based on the tasks to be performed by the database, hence the modeling approach is driven by tasks (roles) of the product (task driven product model).

A set of objects, i.e. products or components, performing a function together is called a system. Components and products may belong to more than one system. System scope and structure must be identified in the product model. The role of each component in the system must also be defined.

A product is multifunctional if more than one function is performed by the product or any of its components. This results in multiple roles or viewpoints from which the product can be evaluated. In NEUTRABAS each function is associated with a system and can be regarded separately. The approach is taken that distinct functionalities will be mapped onto multiple views of the database. Since components can be associated with more than one function and hence system, the database must support multiple streams of attribute inheritance, at least one for each function. This requirement certainly adds complexity to the integration of multifunctional products in a single database.

Another demanding requirement results from the fact that the functional capabilities of a product are often evaluated at several different levels of idealization. Functional property attributes thus cannot always be assigned to the physical product model, but belong to some idealized form of the model. Pipe flow analysis, e.g. may be based on a hydraulic substitution system, while pipe stresses are evaluated by means of some

structural idealization. In consequence, a multifunctional product database must administer several types of idealization of the product model. To maintain consistency simple transformation rules are desired between the physical and idealized version of the model.

In practice, this complex modeling task in NEUTRABAS is approached on the basis of an object-oriented methodology. This facilitates a division of labour and a step by step procedure. The complex product is broken down into physical and functional subsets, i.e, components and systems. Partial product models can be independently defined for each subset. The partial models are later integrated into a coherent product model. To facilitate the integration task, which is of course not trivial, it is a practical necessity to develop a perspective of the scope and structure of the entire product model from the beginning. This can best be documented by general application reference models like the GARM (GIEL 88) in STEP. NEUTRABAS has developed its own concepts for the scope of a product model for ships, which are described in the following sections. In addition, NEUTRABAS also considers itself in good conformance with higher level reference model discussed in the STEP community, e.g, the global high-level model structure proposed by TNO in the IMPACT Project (ESPRIT 2165), Fig. 1.

The following procedure was followed in developing the NEUTRABAS partial product models :

- development of an informal concept of the model by enumeration of the basic information elements (entities) and their attribute sets on the level of shipbuilding, application-oriented terminology ;
- specification in graphical form using NIAM (Nijssen Information Analysis Method) diagrams. This representation is well suited for discussions with application experts and partners. To facilitate the dialogue between modeler and expert the abstract, generic form of the model was illustrated by many hand-made, real world examples ;
- documentation of the model in EXPRESS code by manual conversion and precise formalization of the NIAM models, sometimes assisted by a pseudo-code intermediate version and by project guidelines. At this stage entity definitions with entity names, attributes sets, types and subtypes, restrictions and rules as well as verbal definitions are produced. The EXPRESS application model for ships in NEUTRABAS makes use of existing General Resource Models in STEP for geometry, topology, material properties etc;
- the EXPRESS code is checked syntactically by available parsers, which unfortunately usually lag somewhat behind the current language definition. NEUTRABAS further has developed a Data Dictionary Generator for EXPRESS models, which creates a working space representation of entity structure for reference by the NEUTRABAS Data Management Component and other database functions.

3 .THE SHIP STRUCTURE INFORMATION MODEL

3.1 Scope and organization

The user's representation of the ship hull structure is a model which supports categories of applications ranging from conceptual design through ship operations. Three levels of information are defined :

- a global representation of the ship,

- a spatial organization of the hull,
- a complete representation of the hull structure.

3.1.1 The global representation corresponds to the way the user describes the ship as a complex multifunctional entity. It contains a description of ship attributes for which the "ship concept" can be developed and its characteristics applied to a set of functions that the "product" must perform.

3.1.2 The spatial organization concerns major surfaces and volumes which define the subdivision of the ship into its sub-spaces. This includes:

- referential system, with references for 2-D and 3-D geometry ;
- major surfaces, with attributes such as : geometry, topology, material and application upon them ;
- volume organization, with zones and compartments that include attributes as : type, boundaries, referential, contents of loads.

3.1.3 The hull structure representation is treated with four different levels of detailing :

- Global perception of the structure, that allows global characteristics to be computed and associated to applications of the categories that involve the ship as a product, such as:
 - shapes, weights and volumes,
 - subdivision, cargo and operational capabilities,
 - propulsion, equipment and systems within hull.
- General partition of the structure, that provides specific concepts related to the organization of the structure and to its design and fabrication. It consists of :
 - Major surfaces partition, which includes :
 - joints associated with major surfaces,
 - connections between major surfaces.
 - Structural and spatial partition, which includes :
 - zones in the hull : decks, double bottom, bulkheads, ...
 - prefabrication assemblies or blocks,
 - primary substructures.
- A general structure representation, that provides description of :
 - Shell plating, associated to major surfaces, with :
 - subdivision of shell surface into plates,
 - boundaries and interrelation to internal structure.
 - Stiffening, including connections between elements with :
 - standards for shapes, their arrangement and joints.
 - Connections with plates, flat and curved ones :
 - apertures and other discontinuities, including those for :
 - man access,
 - circulation of fluids,
 - passage for duct-, pipe- and cablework.
- Representation of particular details, including design and fabrication norms, graphic symbols and other conventional or user-related information.

This organization reflects the naval architect's view of the hull structure of a ship as

a product of the design and fabrication processes. It is also valid for future connections to some expert system or other A.I. applications.

3.2 Specification

The Ship Structure Information Model (SIM) is one of several subsystems which together are assumed to wholly describe the ship throughout the product life cycle. This SIM is mapped onto a complete Reference Model which is specified by means of two supplementary information model definition tools : NIAM diagrams and EXPRESS language. This Reference Model represents the knowledge of the expert (here a naval architect or a ship owner) and is developed using an object oriented methodology.

The SIM has been specified and applied to a test section of an actual container ship, and it has been proved that:

- the representation that has been defined is in conformity with the topological representations of STEP;
- the terminology used is in conformity with the one used in STEP and in EXPRESS ; however, an extension of existing STEP/NIDDESC entities had to be made in order to suit the needs of the shipbuilding product during its life cycle.

The NEUTRABAS simplified global NIAM diagram for a ship structural system shows an increase of complexity and number of intermediate entities over the corresponding STEP/NIDDESC global NAIM model (Fig. 2).

The SIM is integrated with other NEUTRABAS models that cover the Spatial Organization Model (SOM) and the Outfitting Organization Model (OOM), to produce an Integrated Product Information Model (IPIM) for ships and similar products.

4. THE OUTFITTING SYSTEMS MODEL

The basic entity in the NEUTRABAS outfitting model is "system". It was chosen to describe the primary purpose for the existence of each product function within its surroundings. For the decomposition of a system into its system components a recursive structure is chosen so that a system component can act as a system in its own right. This implies that one system or system component can play different roles or perform different functions in different contexts. This is a result of multifunctionality. Fig. 3 illustrates the information structure for the example of a distribution system.

This construct serves as a basis for the attributes necessary to carry out and accomplish the different engineering tasks. The properties necessary for the evaluation of the system can be associated the system and/or system components. Due to the embedding of a system in its environment references exist to external conditions like loads, temperatures, pressures etc.

Fig. 4 shows the types of functional systems which are included in the NEUTRABAS Ship Outfitting Model. Each property or function of the system identified here serves as integration point for appropriate attribute sets to describe this property in elaborate form.

The association between systems and their functions is described in Fig. 5. The functions are specified in terms of requirements, rules, laws and regulations. Each function has its own description, as illustrated by Fig. 3. The model records the approval status of systems, components and functional units.

5. DATABASE ARCHITECTURE AND METHODOLOGY

5.1 Introduction

A system architecture has been designed to support the above information models. Before looking at the detailed structure of that system, we first consider some of the criteria it must satisfy.

5.1.1 Neutral

The system should serve the widest possible range of CIM modules : design, production, management, etc , in other words, there should be no bias towards any particular system or application area. Thus, in principle, it should be feasible to develop an interface between any CIM module and NEUTRABAS. This objective is realized by the provision of an open procedural interface, defined independently of any target programming language (MOWA 91).

The architecture is independent of any particular database model or computer system. At the same time, it is able to support all major types of database systems in use today. This is achieved by a system-independent data management component, described below, that separates applications from the specific database technology used.

5.1.2 Dynamic

The NEUTRABAS database can be used to coordinate data shared amongst a large and diverse set of application systems. Thus it must be dynamic, allowing systems to interrogate and up-date the database asynchronously. Moreover, the life-cycle of a ship, at twenty years, is far longer than that of most computer systems and it is likely that a NEUTRABAS model will live through several generations of database technology. Evolution of the model over time and across technologies is therefore essential.

5.2 System Components

The NEUTRABAS system architecture is shown in figure 5.1. It is similar to that developed by the IMPACT Project (ESPRIT 2165). We consider each component of the NEUTRABAS architecture in turn.

5.2.1 Application Interface

An interactive interface or pre- and post-processor must be developed between each application program and the data management component. This interface could be written in any programming language provided that a suitable language binding exists.

5.2.2 Data Management Component

The data management component is the heart of the NEUTRABAS system architecture. The data manager interface, the external view of the NEUTRABAS architecture, offers a dynamic, procedural interface to application developers allowing them to connect their systems into the neutral database. The data

manager kernel is effectively an EXPRESS database : allowing the creation, modification, and querying of entity and attribute instances. Requests received through the data manager interface are reduced to simpler requests that are made in turn through the virtual database interface. The virtual database interface is independent of any specific database technology, being biased towards the capabilities of EXPRESS. It offers a procedural interface at the back-end of the system architecture to which different database systems may be connected.

5.2.3 Data Dictionary

The data dictionary stores the EXPRESS information models in a dynamic form. Thus when a request is received by the data management component, it can check the data dictionary to validate the request, or return the structure of an entity.

5.2.4 Specific Database Interface

Any NEUTRABAS information model must eventually be stored in a database management system. The specific database interface maps the virtual database interface seen by the data management component to the relational, object-oriented, or other database system in which the instance data is actually stored. Each object-oriented, virtual database request is mapped onto one or more requests to the specific database used. In practice, an interface to an object-oriented database interface will be easier to develop than a interface to a relational database interface because of the smaller semantic gap involved. Nevertheless, preference is given in NEUTRABAS pilot implementations of Database Interfaces to relational database solutions because of the currently better availability of mature software tools and because of the existence of SQL as a standard interface for relational database access.

5.2.5 STEP Working Form Interface

Finally a STEP working form interface is being specified to allow NEUTRABAS to use both the dynamic and static forms of STEP file as information exchange media. This will facilitate offline transfer of neutral data.

6. CONCLUSION

The NEUTRABAS Project has developed a basic information model for large, multifunctional systems, in particular for ships with primary emphasis on ship structural and ship outfitting systems. This information model is based on the methodology and general resource models of the evolving ISO Standard STEP. The models are specified using NIAM diagrams for planning and the more formal language EXPRESS for formal definition and syntactic verification.

To control the size and complexity of the model an object-oriented approach was taken resulting in several partial models which will be integrated under the guidance of a global reference model. The multifunctional aspect in the product results in independent models for systems with distinct functionalities, each being associated with its set of functional property attributes. This necessitated applying multiple views to the database with corresponding rules for multiple streams of attribute inheritance.

The models cover a large part of the life cycle in the field of CIME.

The EXPRESS models are the basis for the implementation task and the prototype testing.

The architecture of the data processing system has been defined and the software tools needed to create, populate and use the database have been specified in view of the exchange and storage of information on a neutral format.

Neutrabas has also established a good and effective cooperation with ISO/STEP through the US NIDDESC team and the STEP-AEC committee which are developing the ISO documents.

At this stage, the above mentioned results bring an important valuable contribution in the field of CIME. They are very promising and reliable for a good continuation of the project.

With respect to the product modelling, the integration task of the different models is in progress and has to be continued in order to result in a shipbuilding reference model. Such an integration will ensure a good consistency of the different models. The integration will also bring the benefit of simple and multiple inheritance capability offered by the EXPRESS language. That will lead to modify the models in order to reach models with high level of integration, which are more flexible, can be adapted or modified and can be extended more easily to take into account the multiple views required for the description of the project.

With respect to the data processing system, the retained arrangements should allow to support the multiple inheritances and to store in one database a multifunctional product as complex as a ship.

The prototype will be implemented with a relational database ; however it should allow to support an object oriented database, as soon as an access language is standardized.

The project must still evaluate by tests the practical feasibility of the Neutrabas approach.

As a final conclusion, it must be noted that the ESPRIT II project n2010 NEUTRABAS is in a very good agreement with the views which support the definition of the new ESPRIT III programme. So, it may be considered as an important contribution to this new programme in the fields of CIME and Concurrent Engineering.

REFERENCES

GERARDI, M.L. : Application Reference Model for Ship Structural Systems, Working Draft of Part 102, ISO TC 184/SC4/WG1, Version 4.0, June 1990.

MARTIN, D.J. : Reference Model for Distribution Systems, Working Draft, ISO TC 184/SC4/WG1, Version 1.1, Doc. N.443, December 1989.

GIELINGH, W. : General Reference Model for AEC Product Data, ISO TC184/SC4/WG1, October 1988.

ISO CD 10303-11 : Exchange of Product Model Data-Part 11 : The EXPRESS Language, Doc. N496 of ISO TC 184/SC4/WG1, July 1990.

NOWACKI, H., VOPEL, R., PALMADE, O., MC CLELLAND, A., TORA, C., BRECHE, P. : Data Management Component, NEUTRABAS Del. 1.2.2, February 1991.

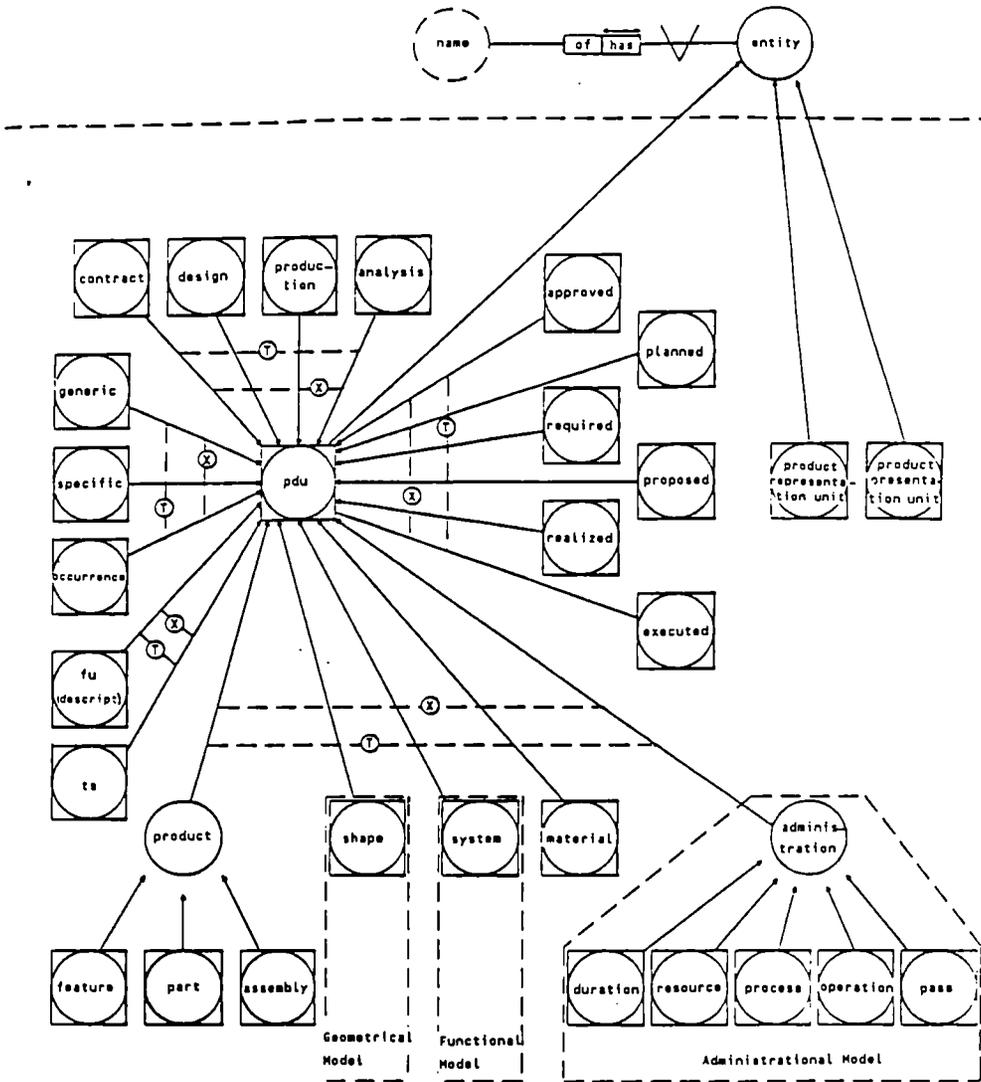


Fig. 1 Global High Level Model Structure (NIAM Diagram),
 Similar IMPACT/TNO Proposal
 (pdu = product definition unit, fu = functional unit, ts = technical solution)

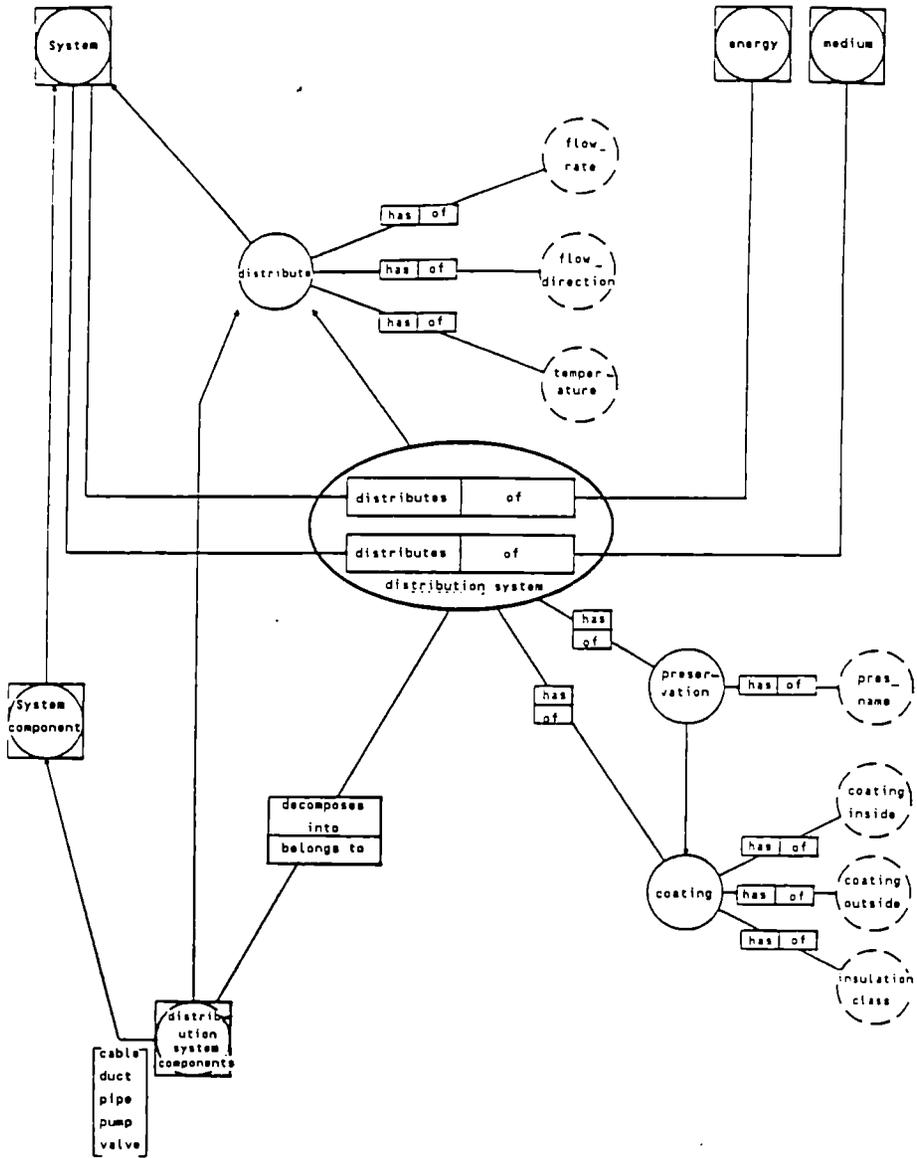


Fig. 3 Distribution System Information Structure

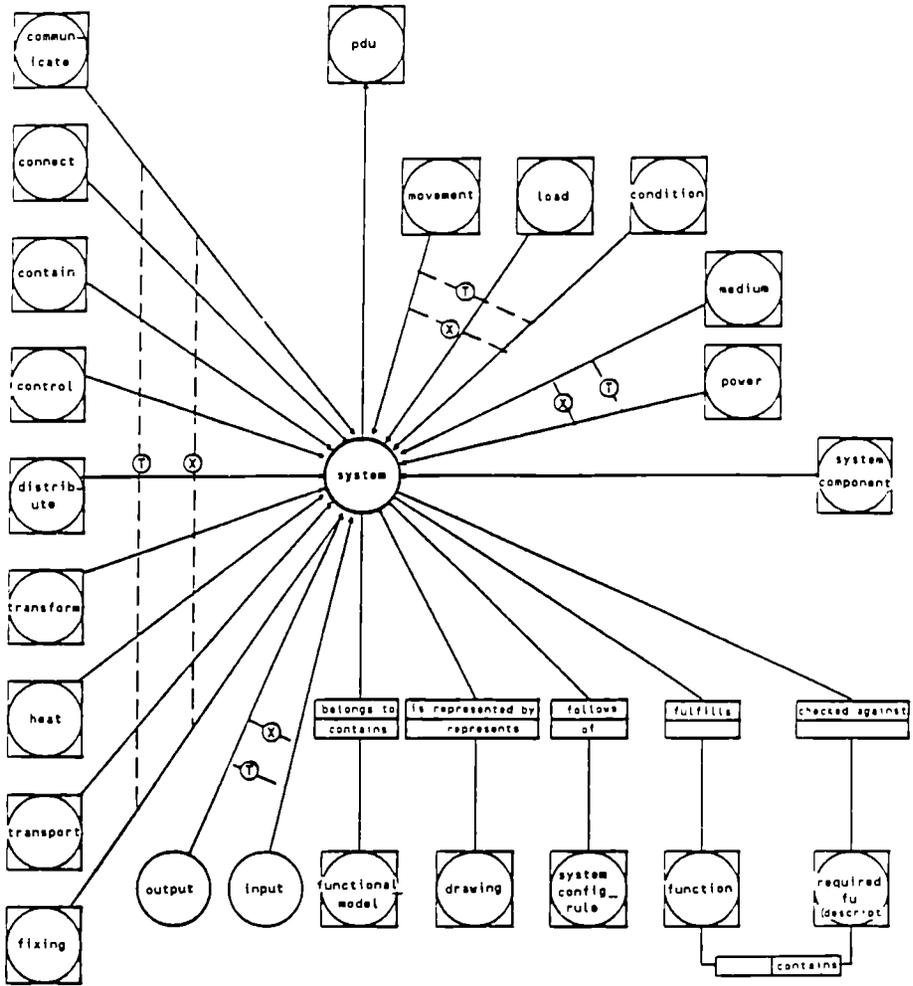


Fig. 4 Functions of Outfitting Systems
 (pdu = product definition unit, fu = functional unit).

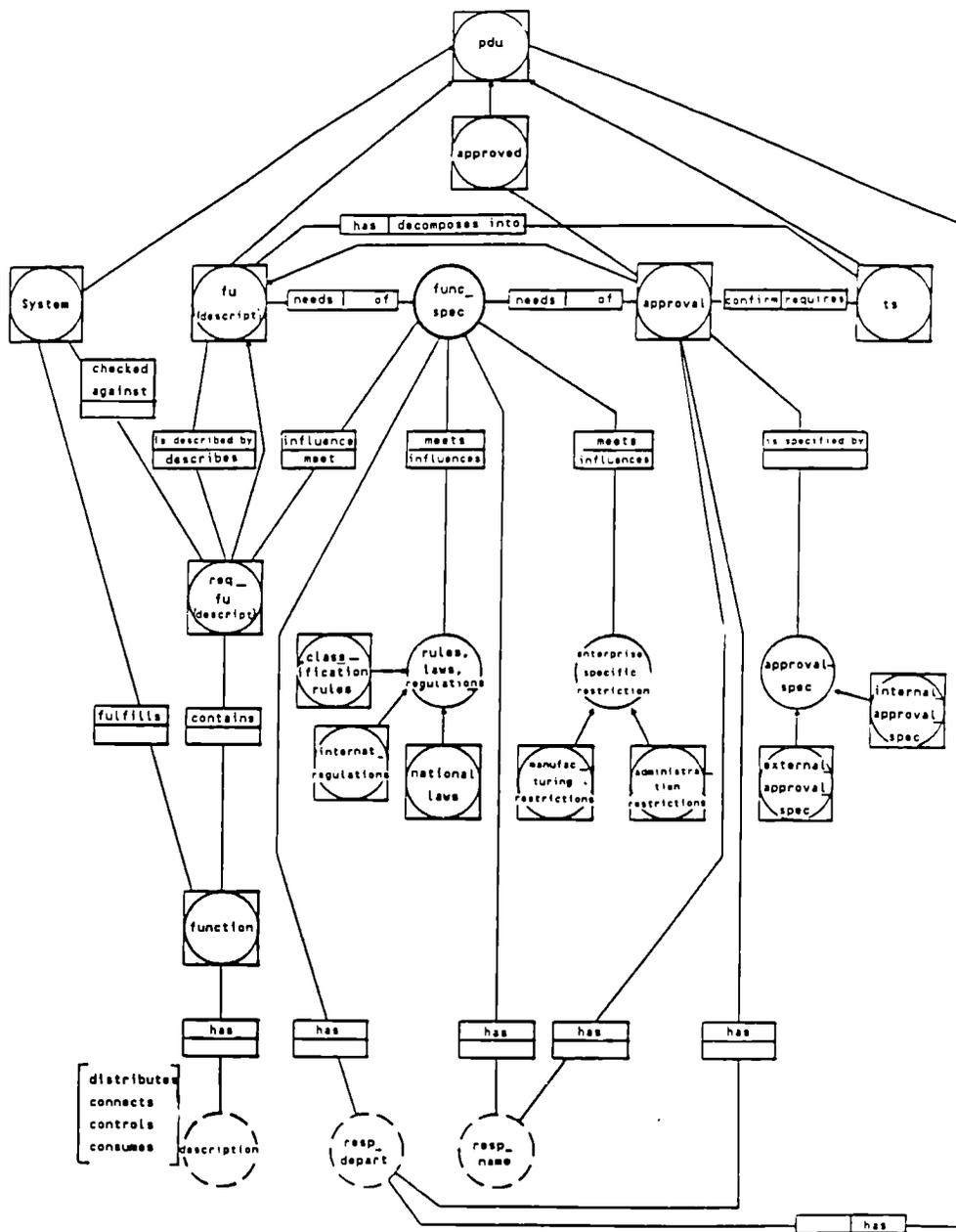


Fig. 5 Modeling of Functions and Specification (pdu = product definition unit, fu = functional unit, ts = technical solution).

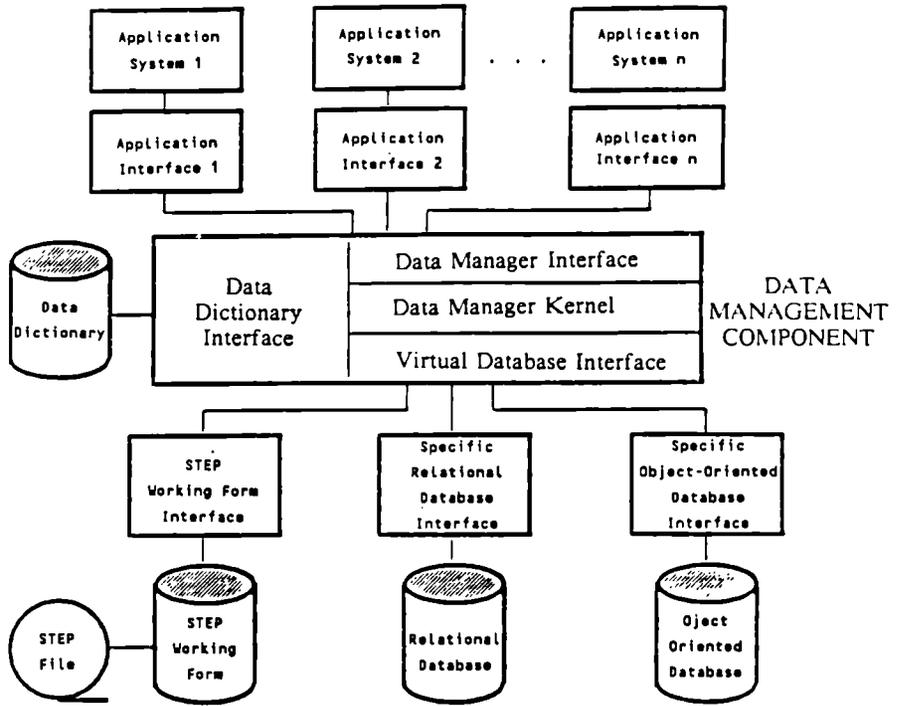


Fig. 6 NEUTRABAS General Architecture

EPIC - AN INTEGRATED ENVIRONMENT FOR PRELIMINARY PROCESS AND CONTROL DESIGN

M. Larintzakis, K. Papafotiou
INTRASOFT S.A
2 Adrianou Str.
GR - 115 25 Athens
Tel: + 30/1-6917763
Fax: + 30/1-6925259

P. Mantzari, A. Efthimiadis
METEK S.A
357 Mesogion Av.
GR - 152 31 Athens

SUMMARY

In February 1989 ESPRIT Project 2090 was started under the title: "Early Process Design Integrated with Control (EPIC)". The EPIC Consortium is constituted of 7 partners: INTRASOFT, PLANET, METEK, MOTOR OIL HELLAS (GR), CITY UNIVERSITY, SAST (UK) and IPL (NL). The aim of the project is to develop an integrated computer-aided process design environment (the EPIC Workbench), enhancing the cooperation between process engineering and control engineering activities at the early stages of the design of production processes.

The project has already achieved significant progress, and the latest version of the EPIC Workbench allows the co-ordinated use of a number of analysis and design tools supporting a complete preliminary process design cycle. In the present phase the project is focused on the execution of the industrial case studies. These will be performed on the Fluid Catalytic Cracker Unit of the Motor Oil Refinery. The objective of these case studies is the validation of the developed tools and algorithms on one hand, and the improvement of the performance of the FCCU through the optimisation of the steady-state operation and the construction of advanced control configurations, on the other. It is believed that a profit of 1-2% will be gained.

1. INTRODUCTION

Current and future computer-aided process engineering systems must support engineers to reach their customers' targets, thus to ensure an improvement of product quality and purity, while reducing the product price and respecting the delivery times. Furthermore, modern plants composed of units with smaller size and medium capacities but with extensive use of recycles and increased degree of energy integration, must operate in a safe and environmentally inoffensive manner.

Therefore, in order to design plants that are safe, straightforward to operate and cost-efficient, an integrated approach is required, combining advanced tools from all the engineering fields involved (eg. chemical process engineering, control systems engineering and optimisation). This requirement presupposes, among others, that the control problem should be studied and solved at a preliminary design stage.

ESPRIT Project 2090 EPIC aims at formulating such an integrated design framework providing a collection of methods, techniques and tools, by developing an off-line computer-aided design system, the EPIC Workbench. This will be achieved by carrying out the following objectives:

- Specify methods and tools for early (preliminary) process design.
- Specify advanced techniques and tools for process control and optimisation.
- Develop the required software infrastructure.
- Evaluate the validity of the developed techniques and tools through the application of the system in an operating industrial plant.

2. THE EPIC WORKBENCH

System Architecture

EPIC Workbench is formed as a series of software layers (see Fig. 2.1), which are described in the following.

The necessity of having enough processing power, enhanced man-machine interface based on the latest standards (X Windows/OSF Motif), and high quality graphics, while keeping to a widely used hardware platform, led to the choice of a SUN Workstation.

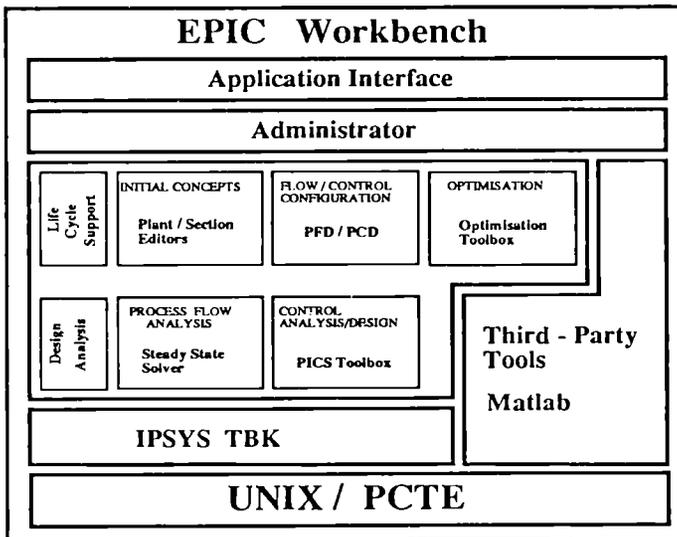


Fig.2.1 EPIC Workbench layered architecture

Nonetheless, the key issue in producing a usable and competent software environment is to ensure an initial structuring of the information, adapted to the needs of users and as close as possible to the physical context of entities involved in the activities of the specific engineering fields. To satisfy this requirement, all types of knowledge in the EPIC Workbench, such as graphical representation of units, lay-out structures, math-

emational models etc., are described as objects or attributes of objects. In the context of the object oriented approach it was found that PCTE covers successfully the database management requirements.

PCTE⁶ (a Basis for a Portable Common Tool Environment) was first developed in an ESPRIT project. It is a hosting structure designed to be the basis for the construction of modern Software Engineering Environments (SEE). Each PCTE based SEE is regarded as an integrated collection of tools and services specific to a particular project life cycle and/or application domain.

PCTE provides a set of comprehensive machine independent facilities. Among these is the Object Management System (OMS). The basic OMS model is derived from the Entity Relationship data model (ER) and defines Objects and Relationships as being the basic items of the environment information base.

Objects are entities (in the ER sense) which can be designated, and can optionally be characterised by:

- a "contents" i.e. a repository of unstructured data implementing the traditional UNIX "file" concept;
- a set of attributes that are primitive values which can be named individually;
- a set of relationships in which the object participates.

Relationships allow the representation of logical associations/dependencies between objects as well as structured information.

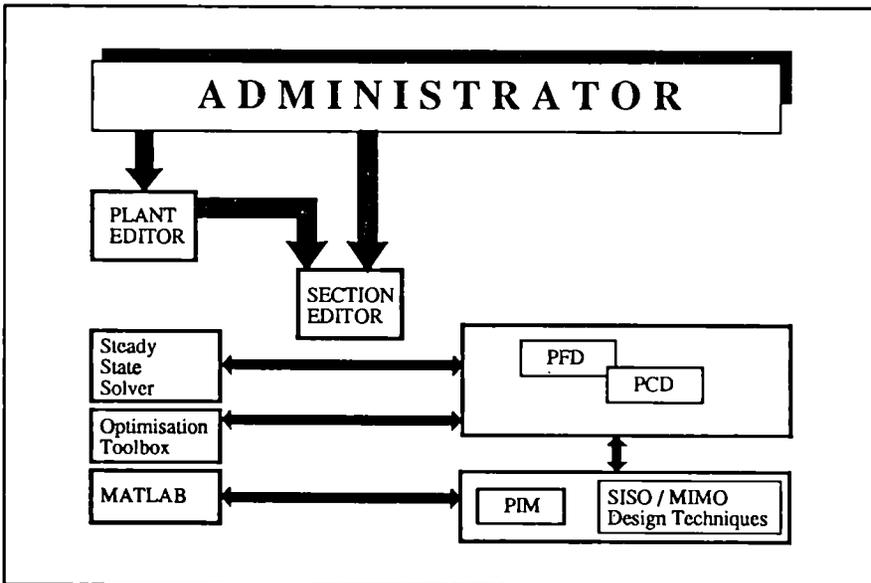


Fig.2.2 EPIC Workbench tools and their interconnections

Built on top of the PCTE is a layer of software, the IPSYS Tool Builders Kit, which is the basic development environment of the EPIC Workbench. IPSYS-TBK⁷ is an integrated collection of generic tools and function libraries aimed at solving the problem of building quality CASE toolsets quickly. TBK addresses issues of integration, compatibility with existing tools, portability across operating systems and hardware platforms, and openness to extension.

Next is the EPIC Workbench kernel layer. It consists of a set of modules supporting all the steps that an engineer follows to design a plant. A short outline of the EPIC Workbench⁸ toolset is given below (see also Fig. 2.2 for the tools interconnections).

- The **Plant Editor** is a tool that supports the determination of the initial system data, such as specifications, constraints, system characteristics etc.
- The **Section Editor** permits the decomposition of the whole plant into independent sections and the specification of the sections interface.
- The **Process Flow Designer** is an enhanced graphical editor, which supports the graphical representation of the process flowsheet and is connected with a number of toolboxes and facilities for the analysis of the plant. Among these are:
 - The **Steady State Solver**, which is a set of algorithms for the solution of the energy and mass balances and the computation of the steady state operating point. The current version is based on the sequential modular approach and uses Mottard's algorithm to find the solution order and the tear streams.
 - The **Optimisation Toolbox** which is a set of algorithms (including novel centralised and hierarchical optimising algorithms) that can be applied to optimise the process operation according to a given objective function and process constraints.
- Last, the **PICS Toolbox** (Process Instrumentation and Control Synthesizer) is a toolkit for the analysis of the dynamic behaviour of the plant, and the design of the control schemes. Focal point of this toolkit is a 2-D matrix, the Process Instrumentation Matrix (PIM). In that, each column represents an actuator and each row a sensor, so that each cell represents an Input/Output combination that may be considered for SISO or MIMO control purposes.

Each cell of the PIM may contain a "vector" describing the process for that particular I/O combination, which has elements like static gain, type of response, delay time, dominant time constant and so on. In addition, the row of the cell represents a sensor, of which the properties are described in a separate vector (e.g. sample preparation and analysis time, if any, accuracy, cost) that is valid for all cells on that row. Each column also has a separate vector, describing the actuator properties (e.g. valve range, type of effective characteristic) for all cells in that column.

Designing the control scheme involves selecting cells that lend themselves to SISO control, and clusters of cells that indicate the need for coupled or full-MIMO control design, and rejecting all other cells. Therefore, the PIM must contain all potentially useful process Inputs and Outputs; which ones are actually going to be used should be decided during the design, on firm grounds. Control analysis/design techniques that have been implemented so far include:

- The Block Relative Gain Array (BRGA) method and a technique based on control-energy indicators for the selection of the most interacting loops.
- The Singular Value Decomposition method (SVD) and the "refined" SVD method.
- Traditional methods for the design of SISO regulators, such as PID tuning procedures, feedforward control techniques or frequency response analysis methods.
- Techniques for approximation and model reduction of distributed parameter and complex models.
- The Characteristic Locus design method of multivariable systems, including also tools for middle frequency compensation and robustness tests.

EPIC Workbench addresses integration from two fundamental aspects: integration by data, and integration by user interface.

Integration by data is the principle behind the familiar ideas of database technology. Tools access commonly shared persistent data whose structure is defined and declared

independently of the tools. This shared persistent data is the basis for integrating the functionality of the tools.

Integration by user interface addresses the powerful integrating principle of consistency of interaction of all tools with the user. Pressing the same button on the mouse or keyboard should have a predictably similar effect, and two things that look the same should have the same properties.

An example of a third-party tool that has been integrated into the system is the PRO-MATLAB toolbox, which is a high performance software package for scientific and engineering numeric computation.

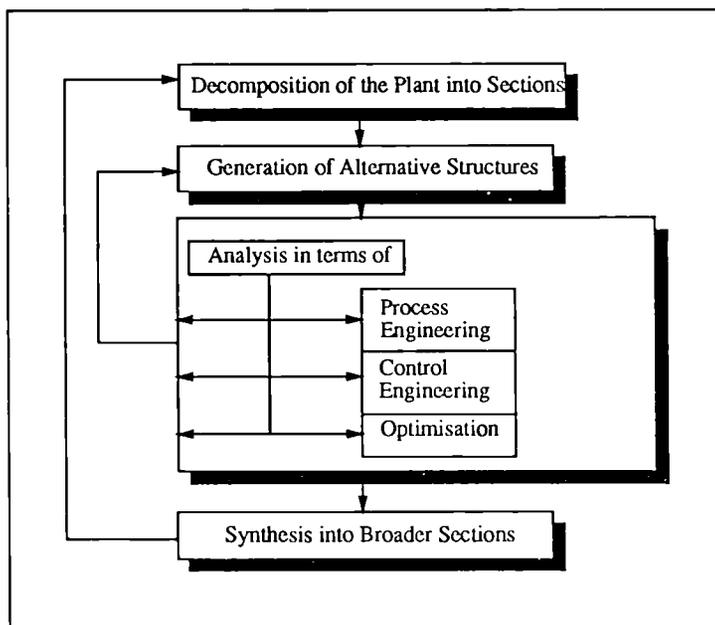


Fig.2.3 Process Design - The EPIC Way

Process Design Steps

The following sequence is a list of steps, which can be taken by a user of the EPIC Workbench (see also Fig. 2.3). Of course, in practice users tend to jump from one step to another, either forward or backward, hence the Workbench should also be usable for permutations of this list. Still, the sequence as shown here corresponds to a systematic analysis of process operability and synthesis of control configurations, whereby it is relatively easy to obtain rough-and-ready results, and more precise results require more effort and time.

- The process designer comes with one or more alternative flowsheets.
- Get static models for process units from the EPIC Model Library and put in the parameter values.
- Run the Steady State Solver to specify alternative operating points.
- Define the optimisation strategy; specify objective function and process constraints.
- Apply the Optimisation Toolbox and determine the optimal operating point.
- Transfer the operating point values into the parameters of the dynamic models and construct the overall dynamic model of the system (internal procedure).

- Call the PICS toolbox and apply several analysis techniques (eg. BRGA, SVD, Control Speed etc), to study the stability and operability of the system and to determine the most significant control loops.
- Design the SISO/MIMO controllers through the application of the available techniques.
- Incorporate the specified controllers in the flowsheet and coming back to the PICS toolbox evaluate the dynamic behaviour of the closed-loop system.

3. CASE STUDIES

Application Domain

Industrial case studies are used to focus, refine and prove various developments during the implementation and testing phases of the EPIC project. Covering the full range of activities addressed by EPIC, the case studies are real-life industrial situations, so that the EPIC Workbench is presented with all the problems that it must overcome.

Nonetheless, to ensure that the case studies can be completed within the project timeframe while meeting the general case study requirements, a single, fairly large process or process section, which allows a variety of designs for the flowsheet and the control systems, and which lends itself to optimising control applications should be used.

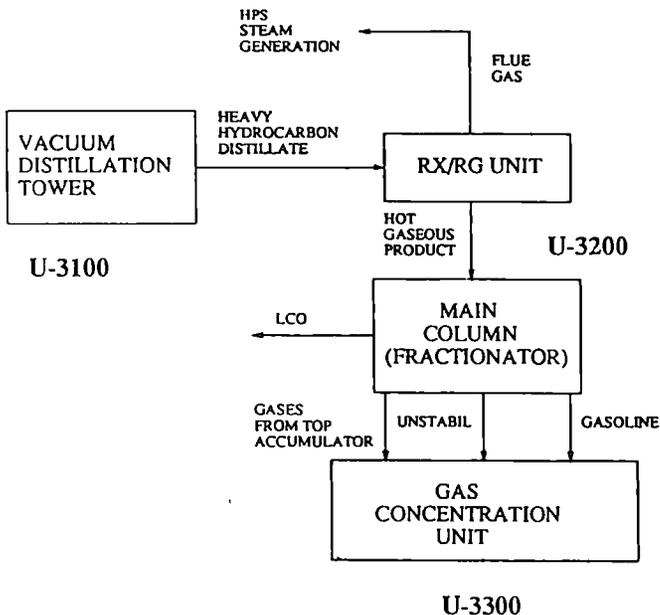


Fig.3.1 MOTOR OIL FCCU

An ideal example was found within the consortium: a Fluid Catalytic Cracker Unit (FCCU) as operated by MOTOR OIL HELLAS⁹. FCC complex is a process with well established and continuously evolved design, allowing a set of alternative optimising control schemes for implementation. It is also a unit with frequently varying operating conditions. This concerns in particular, raw material quality and composition, and product specifications. FCC complex has been erected in 1979/80 and consists of the

following subunits, which are directly interconnected in terms of capacity and operating conditions (see also Fig. 3.1):

U-3100	Vacuum Distillation Unit
U-3200	Reactor/Regenerator Unit
U-3300	Wet Gas Concentration Unit
U-3400	LPC Merox Treatment
U-3500	Gasoline Merox Unit

The most significant and attractive units in the aspect of capacity and yield improvement, as well as optimisation of the operation and control configuration are the Reactor/Regenerator and Wet Gas Concentration Units (U-3200 and U-3300 respectively).

The main bottlenecks that limit the upper bounds of the feed flowrate, and as a consequence, the maximum throughput of these units, are:

- Mechanical constraints imposed by equipment with rotary elements (airblowers, aircompressors).
- The cooling efficiency.
- The main column flooding.
- The regenerator maximum temperature.

These bottlenecks cannot be overcome with the existing control configuration, since this involves only traditional SISC loops with PID regulators (an exception is some ratio and cascade control schemes). In some cases though (eg. antisurge control system), the existing regulators become sometimes unable to control the operation, and then the system switches to manual operation mode.

The approach that is being followed for the solution of the above mentioned problems, is described in the following.

Technical Approach

The only available information for the system was operations data and design parameters. Also, there was not any recorded expertise from the application of advanced control strategies in petrochemical industries.

In order to study the various aspects of the FCCU operation, and to define the context of application of the EPIC Workbench facilities, work has been directed to the following activities:

- Formulation of steady-state and dynamic models. This work includes among others, data collection from the field and usage of parameter estimation and system identification techniques. These models will be fitted in the Model Library of the EPIC Workbench, and serve as a basis for the final study.
- Implementation of the Physical Properties Data Bank.
- Application of the Optimisation Toolbox in order to improve the systems operation under the existing control configuration.
- Application of the PICS Toolbox, to determine the most interacting MIMO and SISO loops and to apply advanced control strategies.

The control strategies are designed in a modular fashion, so that additional control strategies can be added as the needs of MOTOR OIL change. The advanced control strategies selected to provide the maximum economic benefits are in general formulated as follows:

- Capacity utilization control.
- Maximum gasoline yield control.
- Stripping steam optimisation.

- Main column heat balance and product recovery control.
- Improvement of the control system of the gas concentration unit.

These strategies consider operation of the composite system and they are specified into control objectives of regulating or optimising control schemes for each individual subunit.

4. CONCLUSIONS

In line with the objectives of the ESPRIT programme, project EPIC claims contributions to the dissemination of information technology and to the advancement of aspects of engineering and scientific knowledge.

Actual benefits concern advanced software engineering methods and concepts within the environment of practicing process engineers, as well as progress in the theoretical fields of control engineering, optimisation and integrated plant design. Also the progress of Case Study work is well timed as it coincides with the EPIC Workbench reaching a high degree of maturity and integrity.

These achievements encourage the EPIC partners to anticipate significant impact in the field of Computer Integrated Manufacturing for as much as the project produces a useful, open-ended and flexible software package on one hand, and compile valuable information on the development and application of the software for the needs of the process industry, on the other.

REFERENCES

- (1) G. Stephanopoulos (1984), *Chemical Process Control*, Prentice-Hall International Inc., London.
- (2) T. Kailath (1980), *Linear Systems*, Prentice Hall Inc., Englewood Cliffs.
- (3) J.M. Maciejowski, *Multivariable Feedback Design*, Addison-Wesley Publishers Ltd., 1989.
- (4) J. Perkins, *Interactions between Process Design and Process Control*, in: J.E. Rijnsdorp et al. (Eds), *IFAC/EFCE Symposium on Dynamics and Control of Chemical Reactors, Distillation Columns and Batch Processes (DYCORD '89)*, Maastricht, Aug 1989 (Pergamon, 1990).
- (5) P. Brereton (ed.) (1988) *Software Engineering Environments*.
- (6) PCTE, *Functional Specification*, Nov. 1988.
- (7) IPSYS TBK V2.5.1., *Users's Guide*.
- (8) ESPRIT Project 2090, *Deliverable 3.2, "Support Environment - Version 1"*.
- (9) ESPRIT Project 2090, *Deliverable 4.2, "Initial Case Study Specification and Evaluation"*.

HOLOGRAPHIC LABELLING IN CIM ENVIRONMENT

R. Hauck
KRUPP FROSCHUNGSINSTITUT GmbH
Münchener Str. 100
D-4300 Essen 1
Germany
Tel. + +49-20 11 88 31 74

S.J. Halewood
ICI Imagedata
Brantham, Mannigtree
UK-Essex CO11 1 NL
Tel. + +44-20 63 92 424
Ext. 6670

L.M. Bernardo
Universidade do Porto
Laboratorio de Fisica
Faculdade de Ciencias
Praça Gomes Teixeira
P-4000 Porto
Tel. + +351-23 10 290

K. Powell
King's College London
Department of Physics
UK-Strand, London WC2R 2LS
Tel. + +44-71 87 32 836

G. Mazzocchi
Mandelli S.p.A.
Via Caorsana n. 35
I-29100 Piacenza
Tel. + +39-52 37 933

SUMMARY

In modern automated and computer integrated practice, the material logistics in the workshop becomes ever more important. In this context the automatic identification system plays a fundamental role in the control and management of the material flow by allowing the correct and prompt distribution of the relevant material information within the production process. This paper contains the latest results of the ESPRIT HIDCIM project of which the objective is the demonstration system that is based on tags carrying computer generated holograms. The industrial manufacturing test bed associated with the project is described and an outline is given of the integration of the new identification system in the already existing CIM environment; the context of the application is the workshop tool management system.

1. INTRODUCTION

In a computerised industrial environment the identification systems are becoming ever more important. The application requirements are increasing in complexity and difficulty. A new optical technique using holographic identification systems. In this case the information is stored on the ID tag as a hologram, the reconstruction of which shows the coded information. A system based on holographic tags, "holotags", has many simultaneous advantages:

- small and low cost tag with large data capacity,
- immunity to electrical interference during the read and write process,

- immunity to mechanical and chemical interference due to inherent redundancy and surface resistance of the tag,
- wide range of read distances,
- parallel and thus fast read out,
- tolerant and parallel and longitudinal movements of the holotag during readout.

The goal of the ESPRIT 2127 HIDCIM project is to show the feasibility of an automatic identification system that is based on tags carrying computer generated holograms. HIDCIM stands for: holographics labelling techniques for automatic identification in CIM environment¹. The HIDCIM concept is given in the schematic diagram in Fig. 1 showing one cycle of the write-read process.

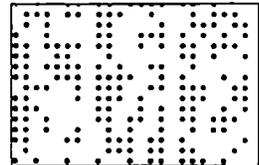
The hardware components of HIDCIM are:

- 1) the test bed: adapters to mount the read station onto an automatic tool loader.
- 2) The write station: comprising a fast PC for CIM net link, bit pattern generation and hologram calculation, and an opto-mechanical scanning system to transfer the data on to the tags.
- 3) The tag material: an optical storage medium (WORM-concept) with direct-read-after-write (without chemical processing).
- 4) The read station: comprising the read head to perform the optical reconstruction of the holotag and a PC for retrieving the information from the reconstructed bit pattern and for the CIM net link.

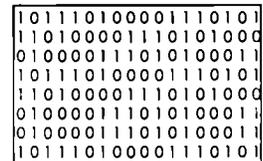
1. The information to be stored in the tag is supplied by a CIM net as a string of alphanumeric characters.



2. Transformation of the input information to a bit pattern, i.e. a 2-D matrix of bits with additional error correction codes.



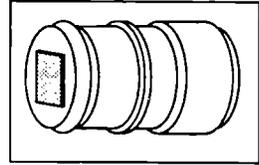
3. Calculation of the corresponding hologram including a Fourier transform, resulting in a 2-D matrix of "zeros" and "ones".



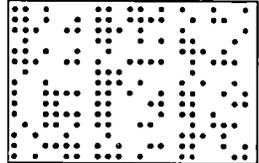
4. Sequential transfer of the calculated hologram pattern to a blank tag by optical writing point by point, forming pits in a plastic material similar to the process of digital optical data storage.



5. Mounting of the holotag (result of 4.) on the tool carrier to be identified.



6. Parallel data transfer from the holotag by optical reconstruction of the holotag, resulting in an image of the bit pattern of 2. on a CCD camera.



7. Retrieving the input information by decoding the CCD image and transmitting it to the CIM net.



Fig. 1: Data flow of the write-read cycle.

2. PROJECT STRUCTURE

The project requires skills in very different fields of technology:

- tool management in a CIM environment to define the actual demands and to open the new perspectives based on the system features,
- communications specific to a CIM environment,
- computer holography to adapt the existing procedures to the requirements,
- information and image processing to code and decode the input information,
- optical design to develop the write and read station,
- chemistry and processing to develop the tag medium.

These technology fields are well covered by the five partners of the HID CIM consortium:

- KRUPP FORSCHUNGSINSTITUT GmbH (KFI)
responsible for the project management, the system layout and for the hardware of the write and read station,
- ICI Imagedata (ICI)
responsible for the tag medium,
- King's College London (KCL)
responsible for the software generating the holograms and for writing the test tags,
- Universidade do Porto (UdP)
responsible for the software for coding and decoding the information including image processing, and for definition and implementation of specific communication protocols,

– Mandelli SpA (Man)

responsible for a specific application in a tool management system, and for definition and implementation of specific communication protocols.

The backgrounds of these partners have guaranteed the technical success of the project. The economical potentials of the partners give a good chance to complete this new ID technology to real applications not only to replace existing ID systems but also to open new concepts in CIM environments based upon the unique system features of the holographic concept.

The tasks of the different partners are highly linked to each other. Intermediate results of one task affects the constraints of nearly all other tasks. Thus a good deal of communication work was necessary to sensitize each partner to the freedoms and the restrictions that the holographic system offers and to ensure sufficient interaction to guarantee the success of the system. The synergy between the partners, which was essential for the project, arose from the expertise and high degree of motivation of each partner.

3. COMPONENTS OF THE HIDCIM PROJECT

3.1 TEST BED

Introduction

For this purpose a machining centre and a part of the Mandelli tool room were equipped in order to integrate the partners' devices and procedures of the holographic identification system. Furthermore, the hostile environment of the test bed allows one to draw as much information as possible in order to extend the utilisation of this technique to a wider range of industrial applications².

Functional features of the application environment

The main features of the application environment can be summarised as follows:

- The Mandelli workshop is composed of a certain number of machining centres whose numerical controls are connected to the Ethernet Network by means of a computer belonging to the VAX (DEC) family. One of the most important tasks of this computer is to manage the tool utilisation cycle in the system.
- The tool utilisation cycle begins in the tool room with the assembly. Once the assembly has been completed the tool physically exists. Then the operator measures the tool offsets to be used on the machine, in respect to the theoretical values, in order to guarantee the desired accuracy of the machining operations. The detected values are directly transmitted from the measuring divide to the supervisor which connects these to the actual identification code written on the holotag of the tool.
- The tool entry into the machine requires the actual tool first of all to be identified. This is done by the read station mounted to the insertion device. The identification code is transmitted to the supervisor.
- Consequently, the supervisor transfers all the relevant tool information to the machine. The machine is now able to use the tool.
- The tool utilisation cycle ends with the tool's disassembly. The tool parts, which can be used again, return to the component magazine, while the relevant identification code and information are cancelled from the data base of the tool management.

The test bed is thus composed of the following (see Fig. 2):

- A Mandelli machining centre and Tool Room station equipped with a new tool input device and holographic read station, integrated within a centralised control system.
- The central control computer supporting the Mandelli tool management system.
- The communications network linking the test bed resources.
- ISO standard tools tagged with holographic labels.

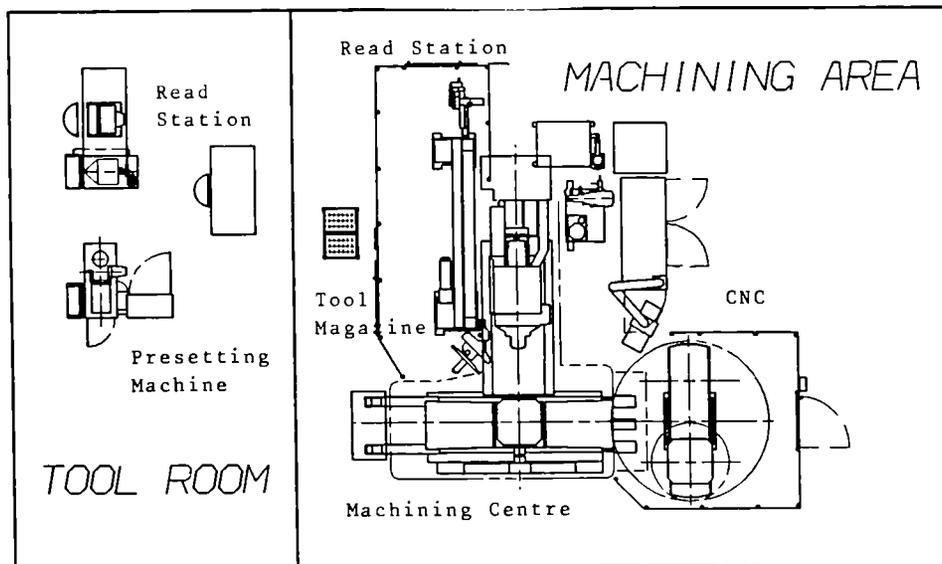


Fig. 2: Test Bed lay-out.

The environment

As previously described, the tool flow takes place from the tool room to the machines and then returned. These are two areas which differ environmentally. The tool room is generally a clean one with acceptable levels of temperature and humidity degrees forming a good working environment with no factors critical for any type of identification system. The machines, however, present environmental features which are more demanding from both the physical and chemical point of view. The main features of this environment include the vibrational disturbances and the relatively high temperatures (70°C) caused by the machines. Moreover, the presence of coolant, which usually consists of mulsifying oils, can act as a dirt carrier, and, due to the chemicals that it contains, can be detrimental to the bonding agents and the tags.

Taking into account these problems, Mandelli and ICI have performed a series on tag materials and adhesives. The results confirmed that the tags meet the preliminary requirement of withstanding the hostile conditions which are peculiar to the Mandelli workshop. The challenge of the environment and problems of integration make the test bed a worthy demonstration of the feasibility of the HIDCIM application.

System specification

Based on the demands, made by the integration of the holographic ID system into Mandelli's tool management, the following system specifications were selected:

information content:	1 KBit,
tag size:	10 mm ϕ ,
active tag area:	3 x 3 mm ² ,
hologram roster period:	3 x 3 μ m ² ,
reading distance:	5 cm,
reading wavelength	830 nm,
reading power:	0.1 mW

3.2 WRITE STATION

The task of the write station is to produce holotags that will contain individual information in code form. The input information, to be stored in the hologram, is transferred to the write station by the CIM net. A software package, written in Microsoft C, calculates the corresponding hologram structure. The main modules of this package are responsible for encoding of the data for the Fourier hologram.

Coding

A string of hexadecimal digits (256 max.) is the information data introduced by the user into the write station. That data is coded and arranged in a bit pattern by a software package integrated in the write station. The patch pattern, whose Fourier transform hologram is stored on the holotag has been designed as a 2-D binary patched grid, surrounded by a metric frame - the synchronisation frame^{3, 4, 5}. Fig. 3 shows one of the patch patterns, that has been studied, containing 6 hexadecimal digits (2, 3 9, 2, 1, 9) repeated 40 times, in accordance with Mandelli's specification. The applicability of the tag to any number of situations is also demonstrated.

The data coding and distribution in the bit pattern was defined in view of the expected problems associates with loss of information in the different steps of information recording, holotag read-out and image acquisition. Each hexadecimal digit is coded in an 8-bit word, having 4 check bits defined by the algorithms of an error detection/correction shortened Hamming code SHC(8, 4)⁶. To improve error detection/correction capabilities, a parity code was further implemented by defining check-parity words with interleaving groups of the 8-bit words. Each bit of the final data set takes the geometrical form of a square patch in the patch pattern. The data is distributed in the pattern with an interleaving format. The error detection/correction potential in the implemented codes, in the decoding operation, depends on the characteristic distribution of the errors in the pattern. Randomly distributed errors deteriorating 2% of the patch pattern are corrected within the correction capabilities of the SHC code with a probability of 93%. Because of data interleaving, burst errors, occupying a localised area of 12% of the patch pattern, can be corrected. The type of error distribution which is more common in the real environment are still to be established by extensive testing on site on a fully networked configuration of the system.

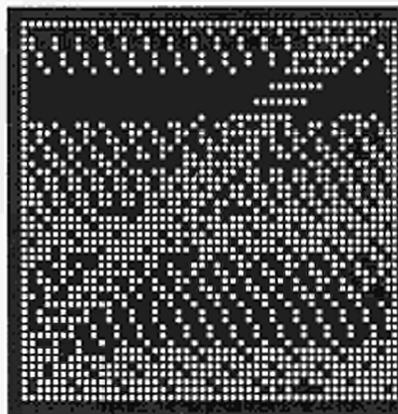


Fig. 3 Patch pattern

The implemented strategy for coding and decoding virtually eliminates the possibility of interpretation of a wrong message as one that belongs to the acceptable lexicon.

Hologram Calculation

Based on the bit pattern, the hologram structure is calculated, mainly by a fast Fourier transform⁷ and a specialised binarisation procedure, to end up with a binary hologram function, see Fig. 4.

The physical hologram will consist of 512 x 512 pixels that can be in two discrete optical states. Ideally they produce an optical phase change of π , that is necessary to obtain bright reconstructed images in the reading process.

The aim of the development was to investigate different coding techniques and to adapt these to the constraints ending in a useful compromise of contradictory requirements:

- high redundancy to allow distortion of the holotag,
- high diffraction efficiency to gain a high signal to noise ration of the reconstructed image,
- fast algorithms to reduce the calculation time and/or hardware demands.

The design time of the holotag on a 33 MHz, 486 PC is less than 30 seconds. This allows an object dependent phase to be designed using a modified Fienup algorithm^{9, 10}, improving diffraction efficiency of the reconstructed holotag is in the range of 24-30%. However, efficiencies of 6% have been achieved on actual holotag material where the absorptive losses and the less than ideal phase change associated with light propagation through the plotted pits in the material reduces efficiencies. Bright, noise free reconstructed images are achieved with efficiencies higher than those required. The diffraction from structures of pits formed on ICI material has been modelled with rigorous electromagnetic theory and predict the results found by experiment. Measurements have also been made regarding the proportion of the holotag that can be destroyed before information is lost. In this way the high degree of redundancy in this holotag has been demonstrated.

Hologon Scanner

The calculated hologram structure has to be transferred to a physical medium. This is done by an optical exposure, point by point, onto the light sensitive recording material of ICI. The exposure causes the formation of pits which is known from processing techniques of digital optical recording.

We have selected a 3 μm roster, thus the size of one hologram will be about 1,5 x 1,5 mm^2 . This hologram is replicated twice in both directions to end up with a total active tag size of about 3 x 3 mm^2 . The replication introduces additional redundancy on one hand, thus part of the tag can be destroyed without affecting the reconstructed image,



Fig. 4: Portion of the hologram structure

and on the other hand it leads to sharper images, which can be more easily interpreted by the read station.

A convenient way to perform the sequential writing on the holotag is to split the two dimensions. In one dimension a fast light deflection system will be used, i.e. a focused light beam moves across the surface of the tag material. During this movement the light source is modulated (switched on and off) to transfer the calculated hologram data of a single line to the tag. In the other direction the tag itself is moved to write line by line.

We selected a hologon scanner as light deflection system consisting of a laser diode with collimator, a spinning disk with a number of identical holographic gratings on it, and a f-q lens for focussing, see Fig. 5.

By spinning the disk, different portions of the grating move through the cross-section of the beam, resulting in different directions of the diffracted beam. By proper design of the grating structure and the lens the light beam is focused at the tag surface and moves along the straight line. The disk is divided into a number of facets, producing identical scans in order to reduce the rotation speed.

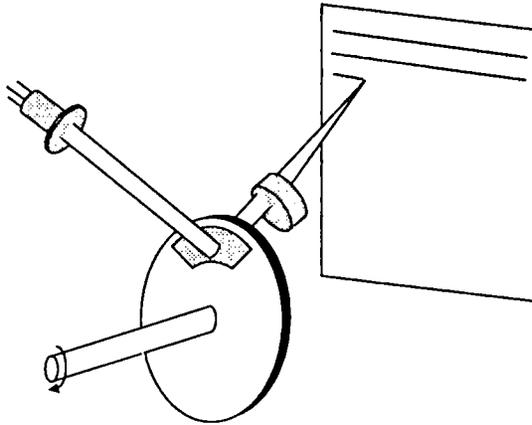


Fig. 5: Sketch of the hologon scanner optics.

In respect to other deflection systems, that are mainly based on rotating mirror, a grating deflector offer the advantage of less sensitivity to bearing wobble¹¹. Furthermore the grating deflector gives the opportunity to introduce optical corrections into the gratings, resulting in an improved performance of the scanned spot (minimising the scan bow, optimising the linearity, optimising the spot shape), combined with a reduction of the number of optical components.

The optical system of the scanner as well as the recording setup for the hologon facets are simulated and designed by a commercial lens design program SYNOPSIS, resulting in the following performance:

- wave length: 832 nm,
- hologon diameter: 80 mm, 8 facets,
- diffraction angle: 56°
- lens: modified Gauss type,
- scan length: 3 mm,

- spot size: $3\ \mu\text{m}$,
- spotsize variation: $0.1\ \mu\text{m}$,
- scanbow: $0.1\ \mu\text{m}$.

The completion of a working write station requires some more engineering with technologies of the state of the art. No problems that need additional research are foreseen.

During the project the test tags were written by a modified XY-stage based plotter of KCL. Different media, besides the target medium were used such as silver halide film, photoresist and chrome on glass to separate effects in the reconstructed images caused by coding and medium.

3.3 LABEL MATERIAL

Over the past four years ICI have been developing an optical recording medium "Digital Paper" for digital data storage. The medium consists of an optically active dye polymer which is stratified onto a metallised polyester base and data is recorded in the dye polymer layer by means of a focused IR laser beam, see Fig. 6. This medium has been developed to give good contrast signals as single data points when read by a highly focused laser beam and can be used to store data at a packing density of 5 MB/cm. These properties were the basis for a medium in which we can store high capacity data in holographic form using a simple, dry and fast process. However, a number of developments are required before we can demonstrate the storages of holograms on "Digital Paper".

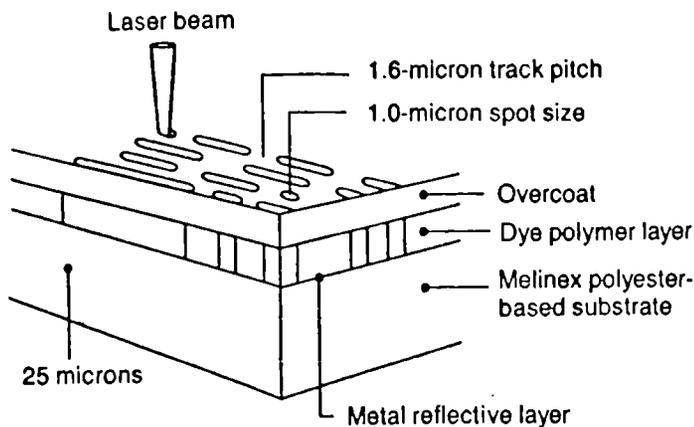


Fig. 6: Structure of the ICI "Digital Paper".

Three major developments were necessary to adapt the "Digital Paper" to the holographic requirements:

- flatness,
- protective overcoats,
- optimisation of diffraction efficiency.

Flatness and lamination development

To reconstruct the information, the holotage is illuminated by an expanded laser beam. Because of the reconstruction in reflection, the holotag acts like a mirror. Any deviations of an ideal plane surface disturbs the reconstructed image. To cope with this the flexible medium is laminated to thick plastic substrate. The lamination technique was steadily improved to yield a surface flatness of 150 nm across an area of 10 x 8 mm² that fulfils the quality demands of standard optics. Measurements were made in an interferometer with computer interpretation of the fringe patterns.

Protective overcoats

The tag media has to be overcoated to withstand the environmental conditions of the machining centre. After two series of tests were conducted at Mandelli, the conclusion was that a thicker overcoat based on the present optical data storage system would provide adequate protection for the tags. A unique brittle failure mechanism was established as the cause of the degradation and work is being conducted in an environmental chamber to reproduce this failure mode. The other system to survive was a laminated polyester overcoat, however, there are serious processing problems associated with this solution. Both of these systems can be written on, however, it is unknown at the present time whether the contrast ratio will be sufficient to produce a high diffraction efficiency hologram.

These tests are still in progress and it is expected to give some more details in the final report.

Optimisation of diffraction efficiency

The "Digital Paper" was optimised for a serial bit by bit reading, whereas in the holographic read out cumulative effects influence the light efficiency. Mathematical models were developed to understand the effects and to conduct the media optimisation.

Subject of optimisation was:

- thickness of the dye layer,
- concentration of dye,
- focus diameter of write beam,
- write energy.

Diffraction efficiencies of 6% are achieved yielding bright and noise free reconstructed images. Test of diffraction efficiency are still in progress in regard to different overcoat in cooperation of ICI and KCL. Holotags can be produced to the dimensional and flatness specification, on a low volume scale, in the laboratory at Brantham.

3.4 READ STATION

The read station consists of an optical read head, that performs the optical reconstruction of the hologram, and a PC based image processing system, that decodes the reconstructed image to retrieve the alphanumeric information.

Read Head

The read head to be mounted on an automatic tool loader of Mandelli is shown in Fig. 7 with the optical system of Fig. 8.



Fig. 7: Photo of the read station.

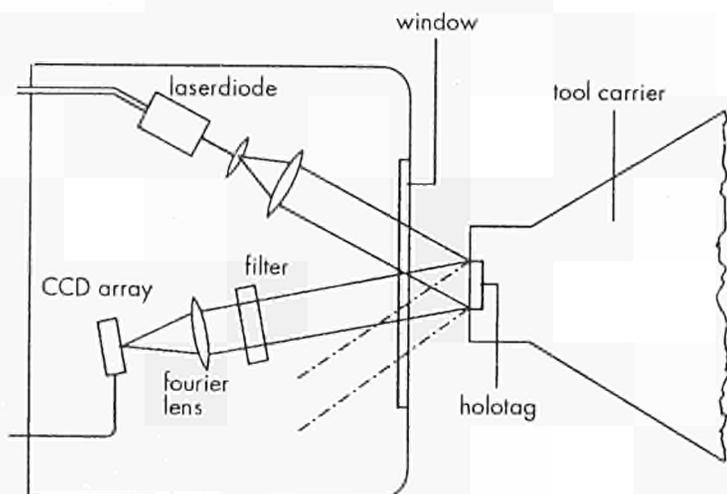


Fig. 8: Sketch of the read station optics.

A laser diode mounted to a collimator illuminates the hologram by a parallel coherent light beam. The diffracted light of the hologram is picked up by an optical system to form the reconstructed image on a CCD-array. The CCD-array is located at the back focal plane of the optical system, in other words the CCD-camera is focused to an object at infinity. In this situation an optical Fourier transform is performed between the planes of the hologram and the CCD-array. We therefore have the shift invariance properties of such imaging systems: if the hologram moves laterally or longitudinally to the illuminating beam, it only becomes weaker when the hologram leaves the illuminating beam. The hologram even may move or vibrate during the read out process. A critical point, however, is a hologram tilt around an axis parallel to the hologram surface. The hologram acts like a mirror causing lateral shifts of the reconstructed image with the danger that parts, or even the total image may fall outside the CCD-array.

A mathematical model of the reconstruction process shows interdependencies of all relevant optical parameters such as wavelength and spectral bandwidth of the laser light, beam diameter, raster size of the hologram structure, size and pixel number of the CCD-array, focal length of the pickup optics, reading distance and tilt tolerance. Some iterative procedure was necessary to come to a design that meets all requirements and constraints. The chosen parameters are: a monomode laser diode with a wavelength of 830 nm, a beam diameter of 3 mm to allow a lateral hologram movement of at least 1,5 mm, a commercial Pulnix CCD-camera with an array size of 8.8 x 6.6 mm² and a cell size of 11 x 11 μm², a hologram raster size of 3 μm and a commercial video lens with a focal length of 500 mm.

This parameter set allows a hologram tilt of 1°. Experiments show that an illumination power of 30 μW is sufficient for adequately reconstructed images, so no problems with laser safety will arise (class I system).

Data decoding

The reading of the holotag by the read head generates in the CCD target of the read head an image of the patch pattern which is digitised and transferred to a frame grabber plugged in the PC computer and the read station. Low contrast and non-uniform, intensity distribution, speckle, spurious reflections, zeroth order noise, distortions, loss of definition and lost patches are some of the aberrations present in the image. The read station is provided with software that increases the image contrast, localises and validates the pattern. Using local algorithms, the developed software reads and binarises the image to extract the data in the presence of a significant amount of noise. After reorganization of the data, error correction, decoding or message invalidation, the information is sent to the user, according to protocols already defined, implemented, and tested in the actual environment network.

Complementary software has also been developed and installed in the read station. It allows the implementation of both the auto mode under central computer management and menu mode under local PC keyboard control. Initialisation, diagnostics, communications protocols and data storage packages are simultaneously installed. Fig. 9 shows, in a schematic form, the structuring of the software packages installed in the read station. The image processing library includes: focus adjustment, gain adjustment, histogram, zoom, overlays, live and snap images with save and load capabilities. Correct threshold and data validation are essential operations integrated in the software. Other programs essential to the project development have been written, but they are not included in the final software version of the reading station package. Developed software is supported and also extends the capabilities of a powerful, data

structured, library package, Software Package for Image Processing (SPIP), developed at ISPRA¹².

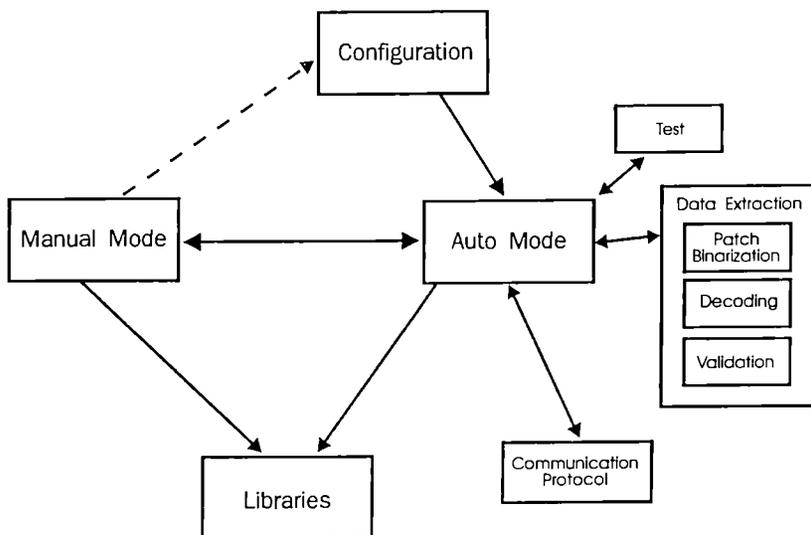


Fig. 9: Structure of decoding software.

4. CONCLUSIONS

The base technology for the realisation of this project on information processing in CIM is optoelectronics. According to a PROGNOS study the market for optoelectronic systems in general offers for the 1990's the prospect of growth rates beyond the corresponding figures for nearly all high technological markets. New developments in this technology have opened up countless niches, though in the volume market seeing robots and consumer goods such as compact discs, Japanese suppliers have long since taken the lead. In military and avionics applications US suppliers form the vanguard.

In the field of civil industrial applications, no similarly pronounced regional supplier structure is yet discernible. European suppliers thus will have the opportunity to gain and exploit market advantages if they succeed in achieving the breakthrough in optoelectronics with new ideas for novel technologies and applications.

The holographic ID system is adaptable to different requirements and comprises favourable features:

- high data capacity,
- low cost tags,
- fast and reliable read out,
- positioning tolerance.

In HIDCIM the application of this new technology in tool identification for machining centers has been explored, but there are also other possible applications, e.g. in assembly lines with mixed products where each basic part, for example a printed circuit board, has a tag with the necessary information for the different parts to be assembled. Compared to bar code, the 1 kbit containing holotags store much more information.

They even store more information than necessary for logistic purposes, where only a few digits are sufficient as addresses for the relevant information in a central computer. Data concerning production or life of the object may also be added to the holotag.

Further application in the electronics industry are in discussion for documentation of parts and subsystems to reduce the volume of handbooks and for data transmission capacities. Here information contents of some 10 kbits are necessary that can be met by the holographic concept.

Coming back to our case study, the results so far achieved are encouraging; the described Test Bed will supply in future months all the required information for the evaluation of the real industrialisation possibilities of the new identification system in all of its different application realities.

REFERENCES

- (1) DUFFY, J.F. and WALD, S. (1990). Holographic Labelling Techniques for Automatic Identification in CIM Environment, ESPRIT 2127 - HIDCIM. Edited Periodic Progress Report.
- (2) MAZZOCHI, G. and WALD, S. (1991). Holographic Labelling Techniques for Automatic Identification in CIM Environment. Proceedings of the 7th ESPRIT CIM-Europe Conference, Turin.
- (3) SOARES, O.D.D., BERNARDO, L.M., MORAIS, F.P., PINTO, M.I. and LAGE, L.A. (1991). Holotags for Security Applications. Proceedings of Holographic Systems, Components and Applications, Edinburgh (to appear).
- (4) SOARES, O.D.D., BERNARDO, L.M., PINTO, M.I. and MORAIS, F.P. (1991). Holotag. A Novel Holographic Label. *Industrial Metrology* 1, 323-341.
- (5) BERNARDO, L.M. (1990). Sistema CIM para Identificação automática usando etiquetas holográficas. *Robotica e Automatização* 4, 61.
- (6) LIN, S. and COSTELLO, D.J. (1983). *Error Control Coding*. Prentice-Hall, NY.
- (7) GOODMAN, J.W. (1968). *Introduction to Fourier Optics*. McGraw-Hill, NY.
- (8) HAUCK, R. and BRYNGDAHL, O. (1984). Computer-generated holograms with pulse-density modulation. *J. Opt. Soc. Am. A* 1, 5-10.
- (9) FIENUP, J. R. (1980). Iterative method applied to image reconstruction and to computer generated holograms. *Opt. Eng.* 19, 297-305.
- (10) FIENUP, J. R. (1982). Phase retrieval algorithms: a comparison. *Appl. Opt.* 21, 2758-2769.
- (11) KRAMER, C.J. (1991). Holographic deflectors for graphic arts applications: an overview. SPIE 1454, San Jose CA.
- (12) CONCALVES, J. G. M. (1989). SPIP - Software Package for Image Processing and Analysis. Technical Note 1 89, 103. Commission of the European Communities Joint Research Center - ISPRa site, Institute for Safety Technology.

ISDN MOBILE TERMINALS VIA INDOOR DIFFUSE INFRARED CHANNEL

D. ROVIRAS, M. LESCURE, C. CHAPUIS
Ecole Nationale Supérieure
d'Electronique,
d'Electrotechnique, d'Informatique et
d'Hydraulique de Toulouse.
Laboratoire d'Electronique,
2 Rue C. Camichel
F-31071 Toulouse Cedex
(Tel: 33 61 58 83 14)
(Fax: 33 61 62 09 76)

R. MESCHENMOSER
ROBERT BOSCH GMBH
Research Institute Communication
Hildesheim
Robert Bosch Str 200
D 3200 Hildesheim,
(Tel: 49 51 21 49 49 99)
(Fax: 49 51 21 49 25 38)

SUMMARY:

The main goal of the ESPRIT project FCPN (Factory Customer Premises Network) is the implementation of a Factory Network with mobile terminals handling capabilities. Mobile terminals will communicate with the Backbone Network via an Infrared channel. The project began in the middle of 1989. The first experimental results on the infrared channel are now available.

In a first part we present the diffuse infrared emitting strategy with its advantage for indoor telecommunications.

The second part will present the technological choices for the infrared channel. These choices are based upon three points:

- The ISDN compatibility,
- The user (factories, offices ...) requirements,
- The state of the art for available infrared components.

The corresponding electrical architecture for Infrared emitters and receivers is then presented with its implementation for prototypes.

The third part of this paper present the first results of the prototype implementation and the total performances that can be expected from the diffuse infrared transmission channel.

1. STATE OF THE ART OF INFRARED NETWORKS

Commercial applications of wireless infrared communications are mainly: Remote sensing, Telephone (1), Point to point telecommunications (2) and Optical networks (3)(4). All these applications are using directive transmissions. Some experiments have been made with diffuse transmissions (5)(6) but today, it has not been any commercial product identify on the market. The objective of FCPN project is to implement an infrared network with diffuse transmission for industrial applications.

2. PRESENTATION AND SCOPE OF THE FCPN PROJECT

In industrial plants, factories and offices the growth of communication requirements is very important. In effect, in the factory, production control and management need a

lot of interconnections between sensors, actuators, industrial robots, measurement units. The computer-based production has created an enormous flow of data-communications between the host management system and production controlling subsystems.

In offices and industrial laboratories the development of computer-based conception has created an important need of data-communications between computer-based terminal stations, host systems and visualisation or printing systems. This increasing data-flow presents data-rates of various types: from the telephone basic service data-flow (low data-rate of 64 Kbit/s) to high volume data-flow like real-time imaging systems (high-speed data-rates of several Mbit/s).

An other important characteristic of user requirements is the mobility of the "telecommunication units". Here "telecommunication unit" can be a standard telephone, a personal computer that needs to exchange files with other terminals, a measurement unit that needs to communicate data to the production controlling system or an industrial robot that needs to give imaging information to an host system. In the paper we will use indifferently the name terminal or "telecommunication unit".

The mobile capability of "telecommunication units" implies a wireless communication between terminals. It is a very important feature because it offers a great flexibility in the geometric configuration of the office or of the production plant. This flexibility implies in effect, a very low-cost reconfiguration terminal topology compared to wired interconnections.

All those requirements involve the implementation of a wireless network with the following characteristics:

- Capability of handling mobile terminals
- Low data-rates to high data-rates
- Industrial environment

At this point a radio-cellular Network could be a possible solution for the first two points. But other requirements like confidentiality and resistance towards high industrial electrical noise are often required by the network users. An infrared transmission channel is then an alternative. In effect, for military plants, nuclear plants and some administration offices the confinement of data communication carriers inside the building is very important for confidentiality. Radio-frequency carriers are not confined inside the building and can be detected outside when infrared cannot be detected.

The insensitivity of infrared radiations against industrial electrical noise is also very interesting for applications in production plants with very high electrical pollution levels.

The main goal of the FCPN project is to implement an Infrared Cellular Network with mobile terminals handling capabilities.

3. GENERAL ARCHITECTURE OF THE FCPN NETWORK

The FCPN Network is based on a high-speed fibre-optic backbone. A host system and an ISDN PABX can be connected to this backbone. Because of the limited bandwidth of the infrared channel a cellular architecture has been chosen for the infrared network. All the infrared cells are connected to the backbone ring via a "Port attachment unit". An infrared cell can be restricted to a room, using confinement of infrared light by walls and ceiling inside a closed area. Figure 1 gives the general architecture of the FCPN network.

Because ISDN is now an industrial standard for data communications it was very important for the FCPN project to be as close as possible from ISDN. In particular it was very interesting to guarantee the capability of interfacing ISDN devices via the

infrared channel of FCPN. As we can see in figure 1 the electrical interface of the infrared transponder is an "S" interface with full ISDN compatibility.

Because of the great variety of applications (in complexity and data-rates) of such a factory network the terminals that are under study in FCPN project are classified in two categories:

- Low-Cost Terminal (LCT): ISDN 2B + D data-rate capability with standard voice service and short data-messages service.
 - High-Performance Terminal (HPT) with processing capabilities (PC-type terminal) and high-performance sensors (cameras...) handling and interfacing capabilities.
- Because of the very high volume data-flow generated by real-time imaging the HPT will have on-board data-processing capabilities in order to reduce this data-flow.

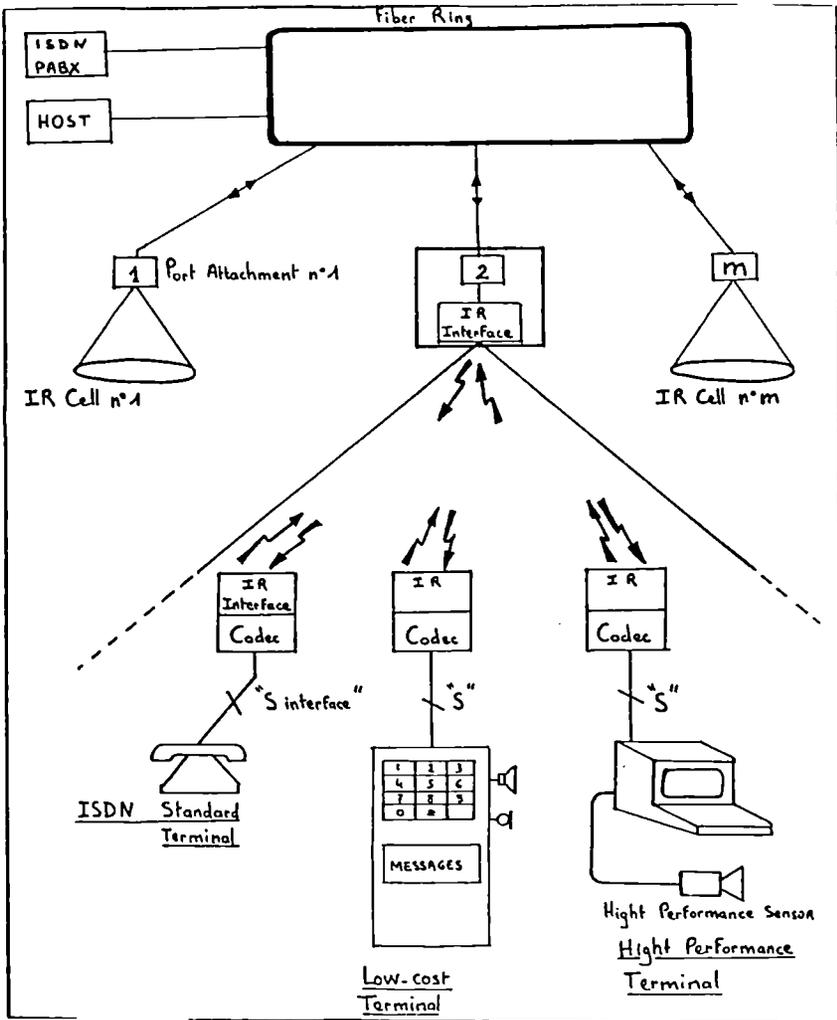


Fig 1: General architecture of the FCPN Network

4. INFRARED TRANSMISSION STRATEGY

For infrared transmissions we can use one two possible strategies:

- **Directive transmissions**, where the receiver must be in direct line of sight from the infrared emitter. These configuration is shown in figure 2.
- **Diffuse transmissions**, where an infrared emitter is lightening the ceiling and the walls of the room. The infrared light is scattered by walls and ceiling in every direction (for most construction materials the Lambertian law is a good approximation of the light scattering phenomenon) creating an entire illuminated cavity. This configuration is shown in figure 3.

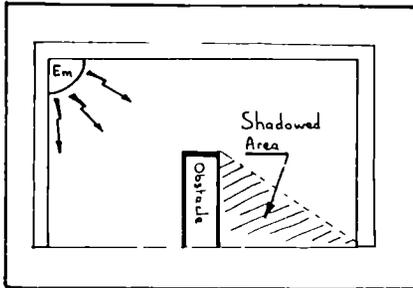


Fig 2:
Directive transmissions.

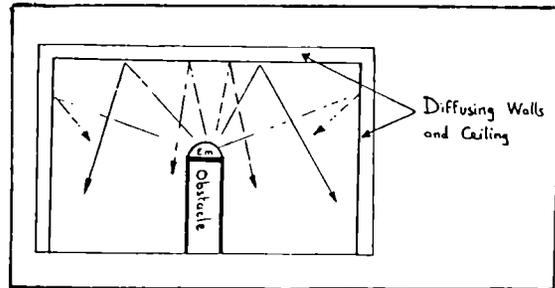


Fig 3:
Diffuse transmissions.

For a good transmission service the infrared channel must have the following properties:

- Ensuring the entire mobility of the terminals in the cell area with poor or no influence of the cell topology.
- Ensuring the non-interruptability of the infrared link by human or industrial activities within the cell area.

In order to guarantee these properties we have chosen Diffuse transmissions. In effect, with Directive transmissions it is possible to have shadowed areas created by the cell topology (obstacles created by furniture or by human activities..., figure 2). When moving in such a shadowed area the session will be interrupted. Another disadvantage of the directive technique is the high dispersion presented by the optical received power. In effect, with such a transmission technique the received optical power is proportional to the inverse of the square distance between emitter and receiver. This will create, in the same infrared cell, areas with low Bit Error Rate (BER) when close by the emitter, and areas with high BER when far from the emitter.

On the other hand, Diffuse transmissions will present a lower dynamic of the received optical power within the entire cell area. In effect, if walls, ground and ceiling are uniformly lightened by the infrared emitter and with the hypothesis of equally lambertian diffusing surfaces, the optical received power is constant for every position and orientation of the receiver. This properties have been studied by GFELLER and BAPST (5). In figure 4 we give some results of computation of the received power for Diffuse transmissions (5) and for Directive transmissions. The emitter is a lambertian Light Emitting Diode (LED), the receiver is a photodiode looking at the ceiling, the ceiling is supposed to diffuse infrared light with lambertian law. We can see that the dynamic of the received power is lower with Diffuse transmissions than with Directive transmissions.

The low variations of the received power in the diffuse case will guarantee a BER quite constant in the entire cell area.

Due to the scattering of light by walls and ceiling the influence of cell topology and human activities is very low, permitting a good quality transmission in every position and orientation of the receiver.

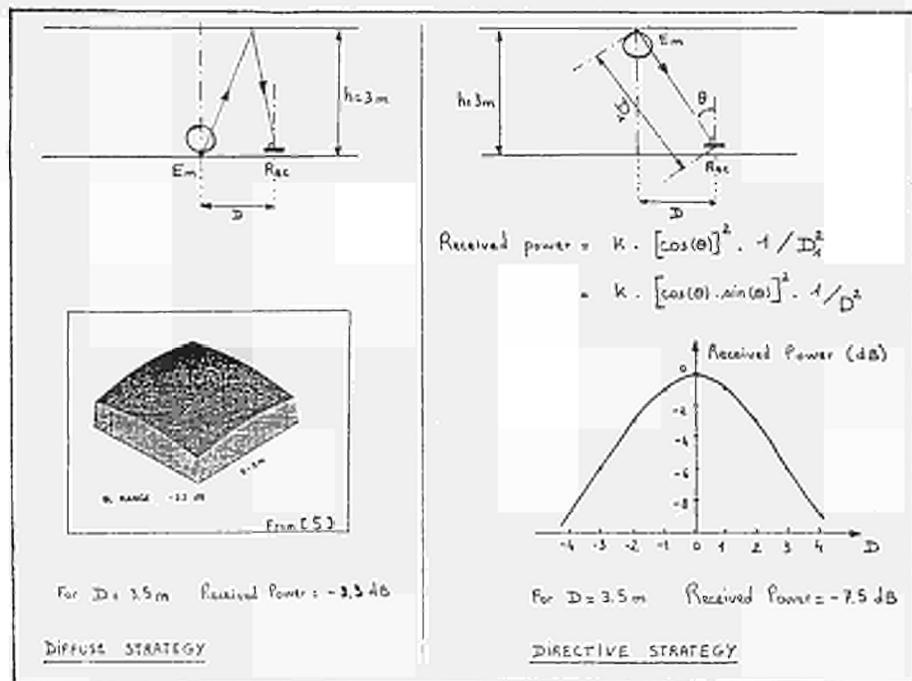


Fig 4 : Comparison of Diffuse and Directive emitting strategies.

5. INFRARED TRANSPONDER ARCHITECTURE

As we said earlier, the compatibility between ISDN products and FCPN network was very important. The "S" standard interface of ISDN (2B + D interface) has been chosen for the electrical interface of the infrared transponder. The electrical architecture of the transponder is given in figure 5. The function of the transcoder (fig 5) is to convert the ISDN frame into an Infrared frame.

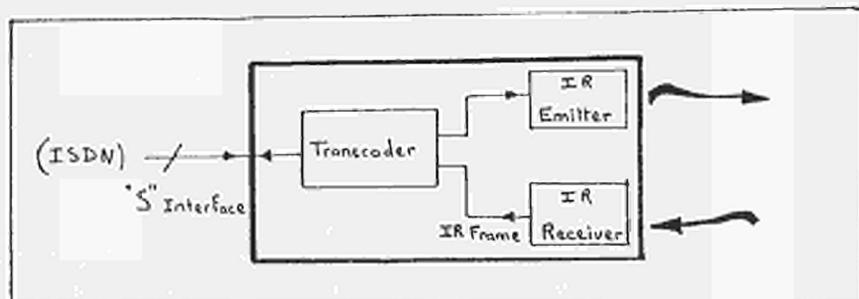


Fig 5 : Architecture of the infrared transponder.

For the full-duplex transmission of the ISDN frame a multiplexing method compatible with the IR channel was to be chosen:

Wavelength transmission multiplexing:

This solution consists in the use of different infrared wavelength for up-link and down-link channels. This solution is not well suited for LED components because these devices have a large emitted spectrum (50 nm at half emitted power), an important dispersion of the central wavelength, and a temperature dependence of this central wavelength ($0.3\text{nm}/^\circ\text{C}$).

This solution is better suited for Laser diodes. The main problem of this solution is the high cost of optical interferential filters and the high cost of laser diode devices. Such a solution, with laser diode devices, could be an alternative for long term opportunities.

Frequency Division Multiplex Access (FDMA):

This solution can be implemented with low-cost, high-efficiency LED components up to data-rates of one Megabit per second approximately. The main problem of this solution is the electrical consumption and the limitation in emitted power due to thermal limitation of LED devices.

Code Division Multiplex Access (CDMA):

This solution is limited by the actual state of the art of LED low-cost components. In effect the increase in data-speed generated by CDMA techniques is not compatible with low-cost, high-efficiency LEDs switching capabilities. This solution is nevertheless under study in the FCPN project for long term opportunities.

Echo cancellation techniques :

Like in two-wires ISDN-type transmissions, echo cancellation could be a possible solution for full-duplex communications. The data-rates on the infrared channel (1 Mbit/s approximately) are too high for low-cost implementation of echo cancellation algorithms. With further developments of high-speed Digital Signal Processors this solution could be an alternative for long term opportunities.

Time Division Multiplex Access (TDMA):

It is the solution that has been adopted for the infrared frame of FCPN. This baseband technique is compatible with low-cost LED devices. Another important advantage of this solution is the possibility of transmitting short-length pulses with low pulse-pause ratios and therefore low thermal penalties for LED devices.

Infrared frame:

The function of the transcoder is to convert the AMI ISDN frame into a frame compatible with the infrared channel. An AMI signal with three levels is difficult to transmit using baseband infrared transmission. In effect, the ambient light is very important compared to the signal (ratio of 10^4 approximately) and with such a DC light component it is very difficult to detect different levels in the received signal. In order to code the

three levels of the AMI code we have chosen a three Pulse Position Modulation (3PPM) coding on the infrared channel. The figure 6 shows the 3PPM code on the IR link.

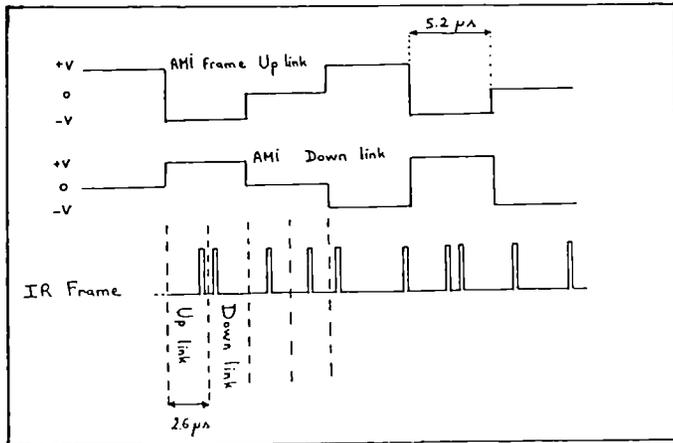


Fig 6: 3PPM coding for infrared link.

As we said earlier it is interesting for consumption and thermal considerations to use short-length pulses. It permits to have very high pulse current in the LED devices and so, very high-power pulses. In effect, for a constant consumption (and so with a constant thermal penalty) we have:

$$\text{Signal} = K_0 * I_{\text{pulse}}, I_{\text{pulse}} = K_1 * \text{Power} / T_{\text{pulse}}$$

with: $K_0, K_1 = \text{Constant}$

$I_{\text{pulse}} = \text{Forward current in LED}$

$\text{Power} = \text{Constant emitted power}$

$T_{\text{pulse}} = \text{Pulse length}$

For the noise we have (considering only shot-noise in the receiver):

$$\text{Sigma}^2 = 2 * I_{\text{dc}} * q * \text{Bandwidth}$$

$$\text{Bandwidth} = 1 / T_{\text{pulse}}$$

with: $\text{Sigma} = \text{standard deviation of input current noise}$

$I_{\text{dc}} = \text{DC current generated by DC ambient light}$

$q = 1.6 \text{ E-19 Cb}$

$\text{Bandwidth} = \text{Total transmission bandwidth}$

When computing the Signal versus Noise ratio we have:

$$\text{Signal/Sigma} = \frac{K_0 * K_1 \text{ Power}}{\text{Sqrt}(2 * I_{\text{dc}} * q)} * \frac{1}{\text{Sqrt}(T_{\text{pulse}})} \quad [1]$$

The first term in expression [1] is constant, so the Signal versus Noise ratio is proportional to the inverse square root of the pulse-length. Using the shorter pulse-length possible, in accordance with low-cost LED devices will increase the quality of the IR link.

But this solution has the major disadvantage of increasing the total bandwidth and having low-frequency components to transmit on the IR channel. In effect if we look at the spectrum of a "real 3PPM signal" (we call a "real 3PPM signal" a 3PPM signal where the pulse length is equal to the bit-slot divided by three) we can see, on figure 7a, that there is no low-frequency components to transmit. On the other hand, if we look at the spectrum of a "modified 3PPM signal" (here "modified 3PPM signal" is a 3PPM signal where the pulse-length is lower than the bit-slot divided by three) we can see, on figure 7b; that there is low-frequency components to transmit on the IR link.

With the ambient DC light it is necessary to filter the received signal with a high-pass filter in order to prevent saturation of the receiver (7). The presence of low-frequency components in the transmitted signal will be an important difficulty for the receiver implementation.

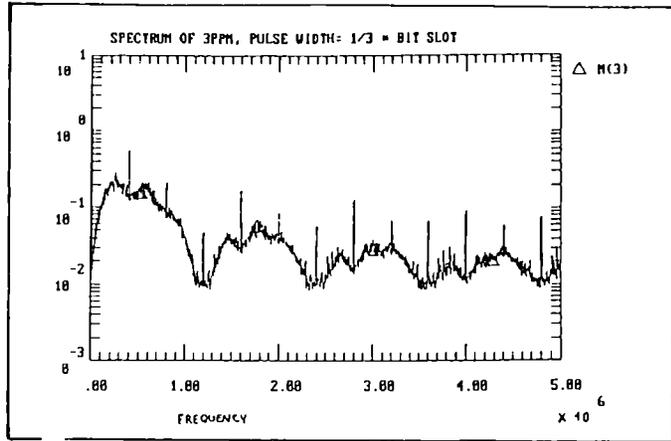


Fig 7a: Spectrum of 3PPM signal.
(Pulse length = Bit-slot/3)

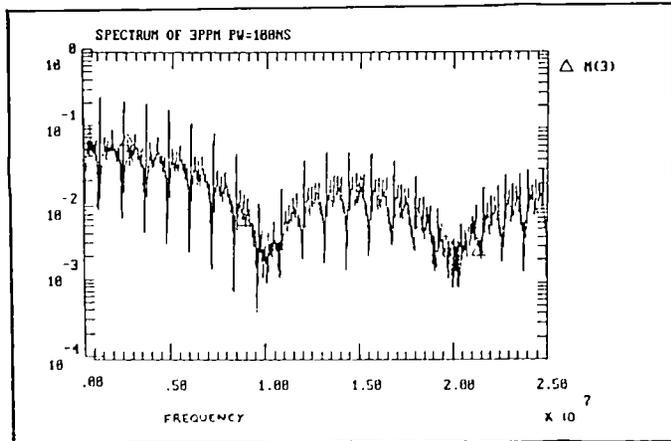


Fig 7b: Spectrum of 3PPM signal.
(Pulse length Bit-slot/3)

The pulse-length that has been chosen for the IR FCPN channel is 100 ns. This pulse length is compatible in width with low-cost, high-efficiency LEDs.

6. INFRARED EMITTER AND RECEIVER IMPLEMENTATION

Infrared emitter:

The IR emitter is composed of emitting-modules. Each emitting-module has one LED driver and six LEDs. The LED devices are Surface Mounted Devices (SMD). For light-emitting components there are additional advantages besides the very small size: the propagation of the light is not hindered by any parts of the case. So the device has an emitting angle of nearly 180° . The emission diagram of such SMD LEDs is approximately lambertian. On the other hand, the cooling of the chip is much better than on plastic encapsulated devices. As efficiency of the LED is decreasing when the chip temperature increases it is very important for the emitter LEDs to have an excellent cooling.

For using the full speed of the LEDs it is necessary to have a driver that can switch a LED pulse current of 1 A within 20 ns. We have tested MOSFETS, Thyristors and Bipolar HF-Transistors. The best compromise between cost and speed is an integrated circuit that is designed as driver for MOSFETS.

The emitting-modules have been build in SMD technology. Each LED driver with TTL input supplies three LED in series. It is possible to select the LED current by variation of the supply voltage of the module. The figure 8 shows a photography of an emitting-module. The board for the LEDs is very thin and has a metallisation on the other side in order to have a good conduction of the heat. For mass production a ceramic with direct bonded chips could be an interesting solution for thermal problems.

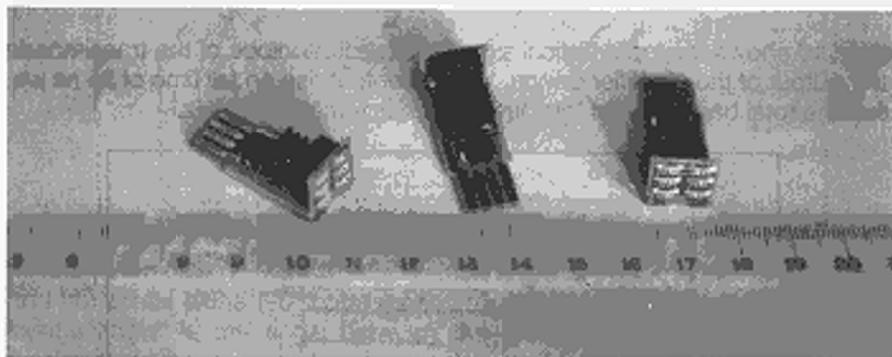


Fig 8: Prototype emitter-module.

Infrared receiver:

With pulse length of 100 ns the receiver bandwidth must go from low frequencies (10 KHz) to 10 Megahertz approximately. The high-pass cutoff frequency of 10 KHz has been chosen in order to reject parasitic light generated by incandescent and fluorescent lamps (7).

The IR receiver has two parts: an optical amplifier and a decider circuit.

The optical amplifier is a transimpedance-type device. The transimpedance structure permits the use of very large area photodiodes with the advantage of high current/voltage gain and high bandwidth. With identical calculations as in paragraph V it can be shown that the Signal versus Noise ratio is proportional to the square root of the photodiode area. Using large area photodiodes is so, very interesting for the receiver.

An SMD implementation of the transimpedance amplifier has been realised in order to minimize size and electrical interferences. The figure 9 shows a photography of the SMD transimpedance. The measured bandwidth of the preamplifier is from 10KHz to 10MHz.



Fig 9: Prototype transimpedance receiver.

The Decider circuit is a high-speed comparator with a TTL output of 100 ns pulses for the Transcoder.

7. PERFORMANCES OF THE INFRARED LINK

Infrared receiver:

Figure 10 shows the input optical signal on the photodiode of the transimpedance and the output of the amplifier device. With an output rise and fall time of 35 ns we can see that the total bandwidth of this device is approximately 10MHz.

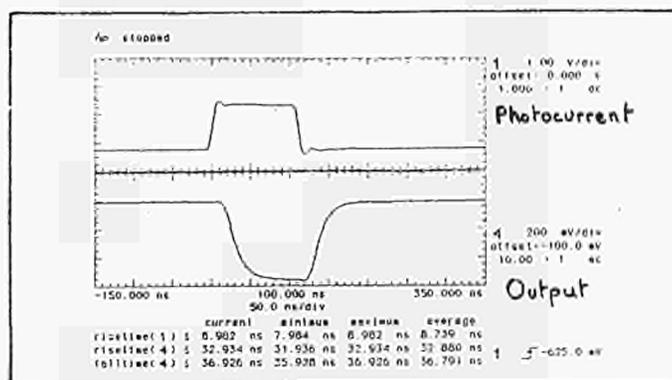


Fig 10: Transimpedance input photocurrent and output voltage.

Infrared entire link:

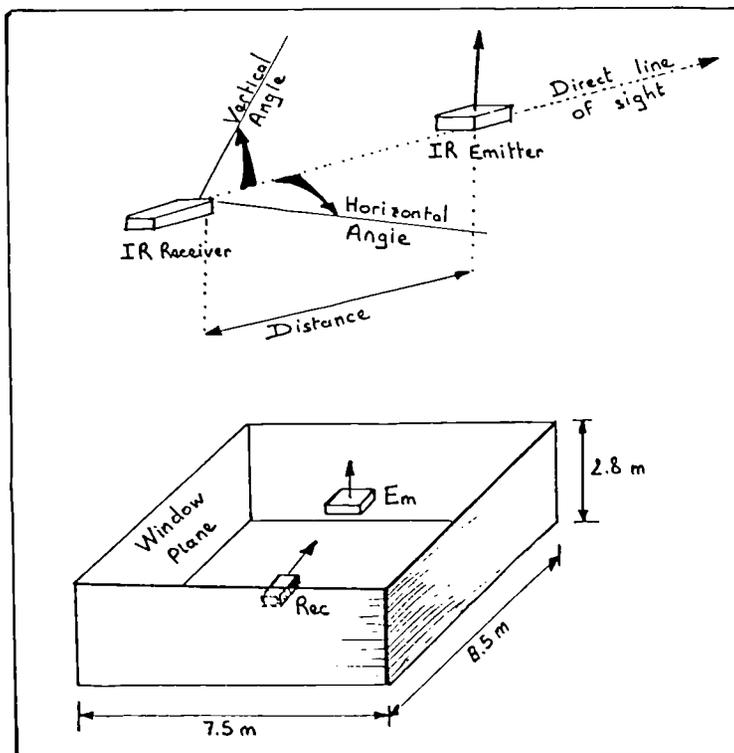
Measurements have been made in an office room. The topology of the room is given in figure 11.

The emitter is built with 18 emitting-modules. The supply voltage for the emitting-modules is 9 volts, corresponding to an average consumed current of 0.5 A. This

corresponds to a LED pulse current of 0.3 A and to an emitted optical-power pulse of 30 mW by LED. So, the total emitted optical power-pulse by the IR emitter is about 3 Watts.

All the BER measurements have been made with a 800 KBit/s binary sequence

Fig 11: Room topology for BER measurements.



The results are given in table 12. The "vertical angle" is the angle between the receiver and the horizontal plane (90° corresponds to a receiver looking at the ceiling). The "horizontal angle" is the angle between the receiver and the direct line of sight receiver/emitter (0° corresponds to the receiver in direct line of sight of the emitter). All the measurements have been made under ambient light. We can see on table 12 that the BER is always lower than $1e-5$ for a distance emitter/receiver of 4 meters and for vertical-angles of 90° . When the distance is increasing the maximum vertical-angle decreases. For a distance of 5.5 meters this vertical angle is limited to 65° . For distances lower than 3 meters the BER is always lower than $1e-5$ for variations of $\pm 60^\circ$ of the horizontal-angle around the direct line of sight and for variations from 0° to 90° of the vertical angle. These measurements show that the dynamic of the received power is quite low with a Diffuse transmission strategy even with great variations of the receiver orientation.

Vertical angle (DEG)	90°	90°	90°	90°	90°	65°
Horizontal angle (DEG)	0°	0°	0°	0°	0°	0°
Distance em/rec (m)	1	2	3	4	5	5.5
BER	<1e-8	<1e-8	2e-8	5e-6	1e-5	3e-5

Table 12: Measured BER for different distances emitter/receiver

8. CONCLUSIONS

The feasibility of a diffuse-infrared transmission has been validated and the first measurements are in good agreement with theoretical results.

The main problem is still the electrical consumption of the emitter. Further investigations are necessary in order to use more efficient LEDs with shorter-length emitted pulses. In particular, special care must be put in the cooling of LEDs devices to prevent thermal decrease of the efficiency.

Laser diodes could be an interesting alternative for long term opportunities. In effect, they permit very short-length pulses and also, a very selective optical filtering of ambient light. The main problem is the cost of laser diodes and interferential-filters which are still prohibitive actually. The development of low-cost laser diodes is nevertheless very important and this alternative must be kept in survey for future infrared links.

Two demonstrators will be implemented at the end of the project:

The first one, implemented in EDF Chatou research centre will test communications between a mobile robot and a host system. The robot will have an on-board high-performance terminal with Image sensors (image compression processing capabilities) and noise sensors (sound processing capabilities for classification and recognition of abnormal noise).

The second demonstrator will be implemented in an office environment with cellular communications for ISDN-compatible hand-held terminals.

9. BIBLIOGRAPHY

- (1) Téléphone individuel à liaison infrarouge, D.ROVIRAS, M. LESCURE, W. SANZ, J. BOUCHER, Colloque OPTO 90 Paris, 15-17 Mai 90.
- (2) Interlaser manual, Zylag research limited, Vol.2, Nov 85, Chap 7.
- (3) Chipnet: an optical network of terminals and workstations, A. Paepcke, D. Crawford, R. Jamp, C. Freeman, F. Lee, Computer Networks and ISDN Systems 15, 88, pp 203_215.
- (4) An atmospheric optical ring network, B. Binder, T. Yu, H. Shapiro, K. Bounds, IEEE Transactions on communications, Vol.38, N°1, Jan 90.
- (5) Wireless in-house data communication via diffuse infrared radiation, R; GFELLER, U. BAPST, Proceedings of the IEEE, Vol.67, N°11, Nov 79.

(6) Infranet: Infrared microbroadcasting network for in-house data communication, F. Gfeller, ECOC 81 Copenhagen, Sept 81, pp.27_1 27_4.

(7) Filtrage des perturbations apportées par le milieu ambiant dans les communications infrarouge domestiques, D. ROVIRAS, M. LESCURE, M. DAURIE, J. BOUCHER, Onde Electrique, Jan 89, Vol 69, N°1.

FATIGUE ANALYSIS AND ACOUSTIC RADIATION PREDICTION IN VIEW OF BETTER CAR DESIGN

J. LEURIDAN / U. VANDEURZEN / H. VAN DER AUWERAER
LMS International
Interleuvenlaan 65
3001 Leuven - Belgium ,

SUMMARY

Acoustic radiation, or noise in general, dynamic fatigue and vibrations are frequently interrelated aspects in the structural dynamics optimization of cars. Realistic assessment of these phenomena is today only possible after development of first prototypes, therefore at a stage where larger design modifications are only implemented at considerable expense.

For acoustic radiation prediction, techniques based on the BEM method are promising but several aspects remain to be improved: the solution to the coupled fluid-structure problem, modelling of specific boundary conditions and proper simulation of excitation. Methodology for traditional fatigue analysis is well developed for predictions based on crack initiation, but less for the crack growth case. Furthermore, the methodology is currently mostly applied at the stage when first prototypes are available. The underlying relation between fatigue and structural vibrations is rarely developed at all, even in the prototype optimization phase. As a consequence, the effect of structural dynamics modifications on the fatigue behaviour is seldom understood.

The ESPRIT project 2486, DYNAMO, develops improved methods for acoustic radiation prediction and fatigue analysis, including the relation with structure vibration models. The technology is implemented in a CAE software architecture that is designed to the now emerging standards for user interfaces and graphics, for UNIX computer workstations.

1. INTRODUCTION

The endurance testing of a prototype of a new passenger car reveals that a body connection of a transmission support frame reveals cracks within design life cycle of the product. What causes this fatigue problem? How is the problem solved most [cost] effectively? Why is it only apparent when test driving a prototype and not predicted from design analysis?

The acoustic radiation predictions for the same car are in reasonable agreement with measurements on the prototype with a specific powertrain and accessories. They are off however for [slight] variations of engine size and interior configurations. Why does the model used for acoustic radiation predictions work in one configuration and does it not predict accurately for any variations? Was one just "lucky" with the good results for the one configuration? How many fixes (and iterations) will be required to obtain acceptable sound quality for all foreseen variants of the car, at what cost?

Many questions, a day to day reality however in design of cars, that specifically concern engineers in the NVH department that are expected to provide answers. Complex performance characteristics such as fatigue behaviour and acoustic radiation, when measured on first prototypes are often found to be worse than predicted from initial design. First decisions on tooling and manufacturing being taken already at the

stage of first prototype availability, design modifications are only implemented at considerable expense. And frequently one recurs to "fixes" that solve the problem at hand, without any reassurance that for the next generation car the original design will be better to start with.

In 1989, the EC approved a 4 year ESPRIT project, no. 2486, DYNAMO, that addresses the development of improved techniques for acoustic radiation prediction and fatigue analysis of cars. The project consortium is composed of four IT companies - LMS International (Belgium, project leader), nCode (U.K.), Straco (France), Trittech Ireland) -, two car company divisions - Porsche (Germany), Fiat Research Center (Italy) -, and three universities - University of Karlsruhe/RPK (Germany), Katholieke Universiteit te Leuven (Belgium), Politecnico di Torino (Italy) -. The multi-MECU project started in May 1989, and has following principal objectives,

- to extend fatigue analysis by life time prediction based on crack growth, possibly in combination with crack initiation, and to develop a systematic approach for relating dynamic fatigue problems to structure dynamics (such as modes of vibration) and therefore possibly developing solutions to fatigue problems from structural dynamics considerations,
- to extend acoustic radiation prediction based on BEM for improved, more reliable applicability to interior noise prediction of passenger cars, and this by including specific boundary conditions (such as damping layers), as well as by use of more modular solution techniques, for the coupled fluid-structure problem, and for the efficient prediction of variants of a car design,
- to implement all enhancements based on an open system architecture, with open interfaces to existing technology (such as FEM pre-/postprocessors), with a link to CAD through technical modelling, and adhering to standards for user interface design and graphics on UNIX computer workstations.

The paper reviews first the state of technology that DYNAMO starts from, and continues with a discussion of the development approach and progress over the first project year.

2. BASELINE AND RATIONALE OF DYNAMO

2.1 Fatigue Analysis

Fatigue may be considered to be made upon the processes of crack initiation followed by crack growth terminating in structural collapse. The local stress-strain (LSS) approach was developed in the early sixties to address the problem of crack initiation [1]. At the same time, the concepts of linear elastic fracture mechanics (LEFM) were established to account for crack propagation [2].

Today the LSS approach is highly computerized and is widely accepted by the engineering community, particularly in the ground vehicle industry, as a bona-fide method of fatigue life estimation [3, 4, 5]. The LEFM approach has not reached the same level of computerization or standardization, although it is widely used in the aerospace industry. Currently attempts are being made to combine these approaches into a unified whole which will be able to predict component life from initiation through propagation to failure [6]. The benefit of this unification will be that a clearer understanding of the fatigue process will be achieved for any given situation.

Fatigue arises from the dynamic loading imposed on a structure or component. The response to dynamic loading, forced or resonant vibration, can be characterised by a series of strain excursions which together comprise a time series. These strain-time

histories are measured from prototype structures or final products in the service environment, and are then used for fatigue life assessment. Given a representative measurement, the accuracy of these predictive techniques for prototype assessment and failure investigation have been well proven. However, a design change will necessitate the production of the next generation of prototype and re-measurement in order to validate the change, primarily because the design modification may affect the structure dynamics. Such effects can not be predicted with current fatigue analysis they enable only to predict the effect of variations of material and geometry, specifically than stress concentration factors, on the expected life, assuming that the loading (strain-time history) would stay identical.

The study of variations of material and geometry on the expected life has its value for understanding the sensitivities of a particular fatigue problem to those parameters, but required changes to solve the fatigue problem may not be implementable for either manufacturing or economical reasons.

Since the strains in a structure are linked to the forcing and resulting response, it should be possible to develop a relation between strains and the structures dynamics. The effect of structural changes on a strain-time history (measured for a given state of a prototype), and therefore on the expected fatigue, can then be predicted. This would give a new dimension to assessing possible solutions to a fatigue problem. It is a main development objective of DYNAMO. Figure 1 illustrates the components of the fatigue analysis process as viewed in DYNAMO.

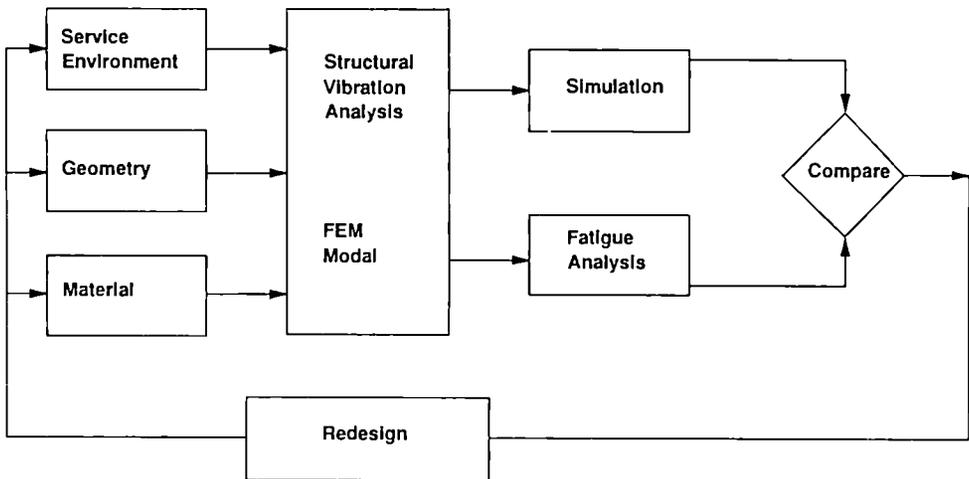


Figure 1 Integration of Structural Dynamics in the Fatigue Prediction Process

2.2 Acoustic Radiation Prediction

In mechanical design, the analysis of acoustic radiation fields (interior/exterior) is perceived as complex and difficult to be realized with any degree of reliability before any prototypes are available. At this stage, fundamental changes can frequently not be

performed on the product, and costly remedies may have to be chosen to meet the required sound power levels.

The trend in automotive industry is to build lighter constructions for reasons of performance (acceleration, braking distance) and energy consumption. This puts an increased demand for proper prediction of acoustical performance. The need for tools which can predict sound levels or other acoustic parameters in the design and early on in the prototype phase is therefore becoming increasingly important. Today, automotive engineers have not only a hard time to predict interior-noise levels for a next car design, they often fail to understand and to predict the changes in noise levels of an existing prototype for design changes or other engine types.

Methods to predict sound power levels or radiation properties for a specific structure with a certain loading, do exist, but these are nowadays mostly used in special circumstances [7-10]. FEM techniques can be used to some extent, but predictions have generally been found to be in poor correlation with measurements. Boundary element methods (BEM [11- 14] have been used for some years, but their application is today generally restricted to specific environments and needs of the developers, and dedicated end-users: underwater acoustics, aerospace applications, in military environments. Some universities developed their own BEM, more directed to air acoustics. Although the BEM lends itself to be coupled with FEM for a global vibro-acoustical approach including the sound fluid interaction [15-17], not many BEM codes include this global approach in the product.

The use of radiation models is certainly not common practise in the automotive industry. If used at all, they do not allow to predict the impact of small modifications (such as change of engine type in otherwise identical car) without a mayor re-analysis of the model, nor can they predict expected noise variances caused by the production process. Current BEM analysis methods lack a capability to develop sensitivity of sound field parameters to structural parameters based on the BEM model for a given state of the product.

BEM calculation programmes draw on results from FEM analysis (static, dynamic) or/and results from structural dynamics tests, and as such are very much interrelated to the analysis of vibrational behaviour. As various types of codes are used by different departments in one company for the structural analysis and testing work, an open system approach is in order to develop an optimal analysis tool for solving vibro-acoustical problems. Actually the various analyses are optimally driven from the underlying CAD model, complemented with the necessary technical features. Blatant design errors should then be detected early in the design. A similar need exists for engineers involved with structural testing and prototypes optimization. The cumbersome procedure of making several slightly different prototypes to obtain best solutions by trial and error could be streamlined by the help of computer simulations of model variations prior to actual implementation of such hardware changes on prototypes.

The development of a global vibro-acoustical approach to solve acoustic radiation problems, with a capability of sensitivity analysis for more efficient application, are also major DYNAMO objectives.

An other interesting development, but focused on a fun experimental approach to treat vibro-acoustical problems, is realised in the BRITE project 2319, "Optimization of Noise Control Measures in Complex Lightweight Sheetmetal Structures by Using Energy Flow Analysis". This project has developed an approach to apply statistical energy analysis (SEA) by measuring all relevant parameters in situ, reducing therefore the amount of empirical data that are required [18].

3. APPROACH AND FIRST RESULTS

3.1 Fatigue Analysis

Including structural dynamics in the fatigue analysis process implies establishing a link between vibrational behaviour and the load data, as illustrated in Figure 2. Fatigue analyzers, whether based on crack growth or crack initiation, will use as input material data, geometry information and load data in the form of strain-time histories. The latter can originate from testing (operating responses), or possibly from dynamic analysis and simulation programmes. Assessing the influence of structural dynamics on a fatigue problem can be done by an analysis of the load data in function of vibration characteristics; e.g., what is the contribution of a particular resonance to the load data, and therefore on the predicted fatigue, or what is the [simulated] effect of modifying a particular resonance frequency and associated mode shape on the load data, and therefore on the predicted fatigue? For dynamic fatigue problems, the solution could indeed be realized by structural modification remote from the location where the fatigue is apparent (crack initiation) if the modification affects the structural dynamics in a manner which is favourable for the problem at hand. Remark the vibration characteristics can be determined from FEM analysis, or from laboratory testing.

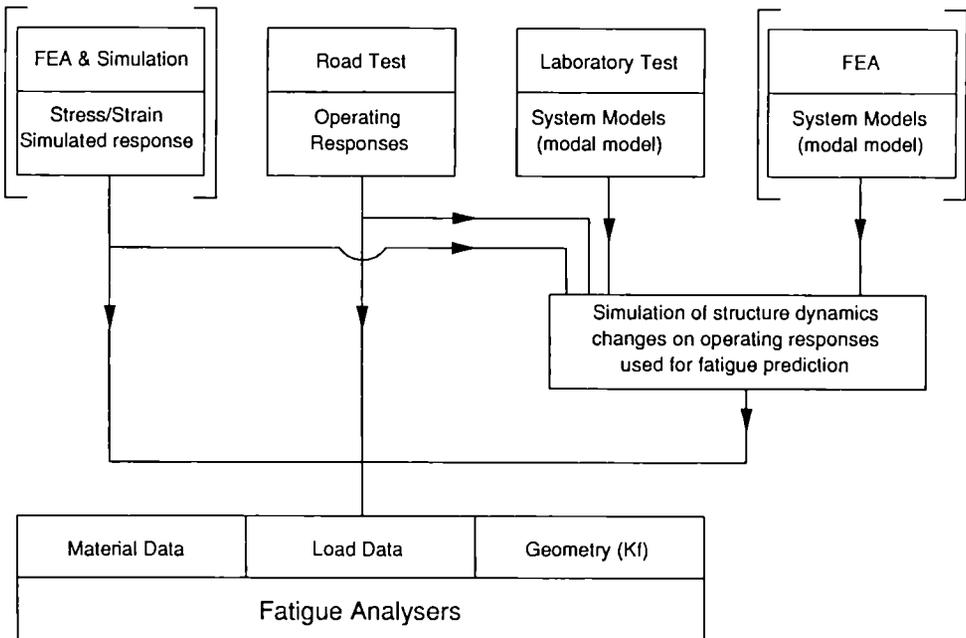


Figure 2 Interaction of Structural Dynamic Analysis with Fatigue prediction

Figure 3 illustrates a typical scenario that would be used for the analysis of a dynamic fatigue problem. 'Road Tests' are executed with both response and strain measurements. A modal model is identified through a 'Laboratory Test'. 'Correlation Analysis'

- which can take the simple form of comparing frequency data - helps to identify which modes are excited during the road test. This information is then used to filter -'Frequency Modal Filter'- the strain time histories for frequency components around such modes. The processed strain time histories are then used for fatigue analysis. Comparing these prediction with prediction on the original strain time histories identifies mode related fatigue. After actual hardware modification, a sine dwell type test at the original trouble frequencies can be used to simulate the effort of actual improvements.

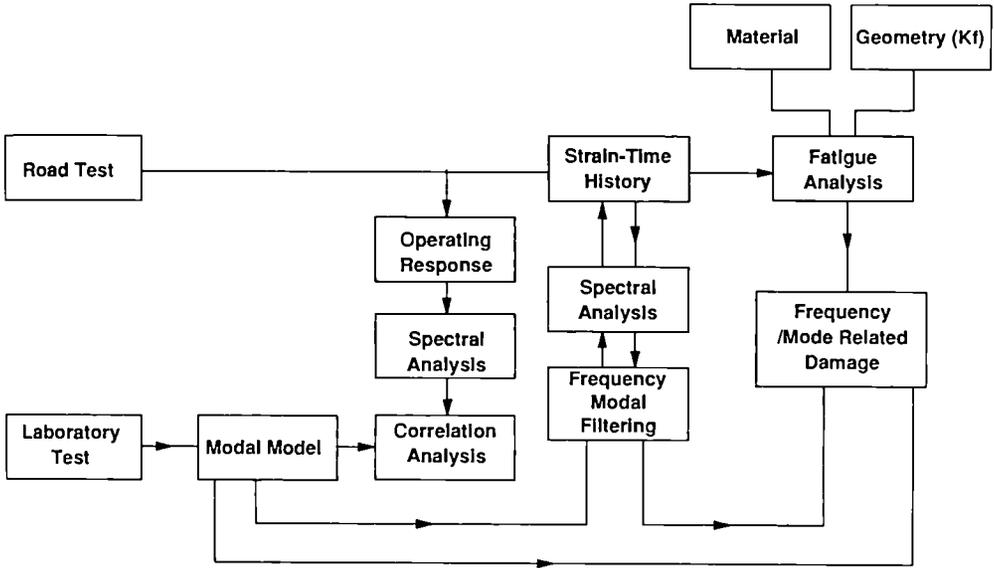


Figure 3 Experimental Analysis of a Dynamic Fatigue problem

This scenario is a principal path to analyze the dynamic fatigue problem and its sensitivity to existing structural dynamics components, but does not enable the prediction of change of fatigue life under certain structural modifications. Hereto, a model needs to be developed that links strain-time histories to the structural dynamics. The analysis procedure in development for this in DYNAMO is illustrated in Figure 4.

The aim of the procedure is to simulate the effect of changes in structural dynamics on the predicted fatigue. Like the scenario outlined in Figure 3, it uses data from road tests, spec. strain-time histories, as well as a structural dynamics model. The procedure is being validated based on experimental procedures to develop the structural dynamics models.

The procedure requires first a structural dynamics model to be developed. The model should include vibration responses as well as strain responses, related to a set of inputs that are relevant for the problem at hand. These models are developed as multiple input multiple output transfer function models, indicated as $[H_y]$ and $[H_e]$ [19]. The strain measurements should be measured at locations that correspond to the locations used to develop the strain-time histories.

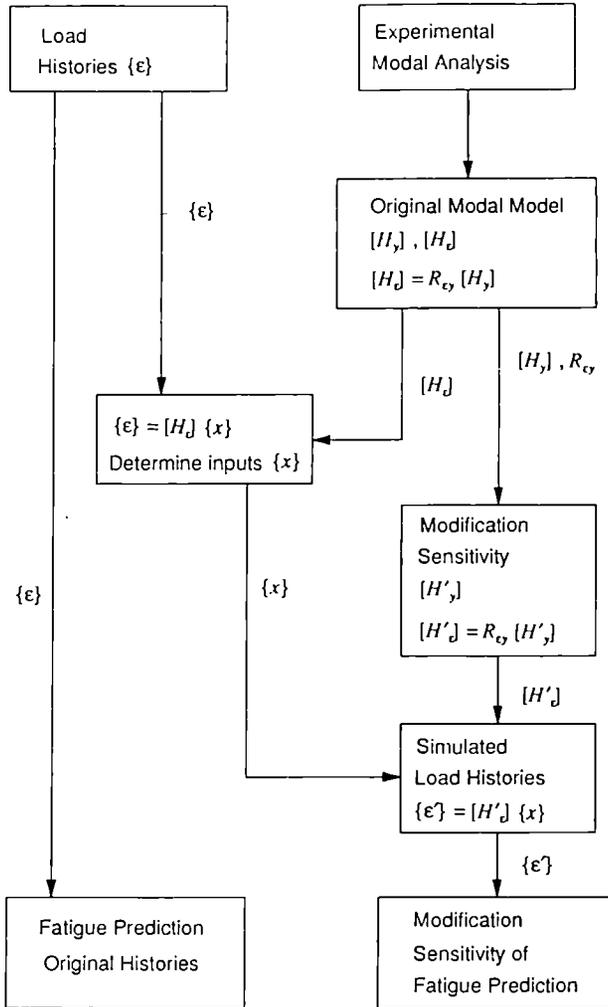


Figure 4 Simulation of Changes in Fatigue Behaviour Due to Changes of Structures Dynamics.

Next a relation is to be developed between $[H_e]$ and $[H_y]$, symbolically indicated by $R_{ye}()$. There are different ways to establish such relations, which are discussed in ref. [20]. The effect of modifications can be implied on $[H_y]$, resulting in $[H'_y]$, and then through $R_{ye}()$ be applied to $[H_e]$, resulting in $[H'_e]$. This is valid provided that $R_{ye}()$ does not change with the applied modification, which is the case when the structural modifications do not affect the local displacement-strain relation. Possible criteria to assess this are currently being studied; one promising method would be to use the Modal Assurance Criterium (MAC) between the structure modes before and after the modification [20].

To simulate the effect of the structural modification on the load data, an inverse force calculation is done based on the load data $\{e\}$ and using $[H_e]$; this yields a virtual set of inputs $\{x\}$ at the locations that are used during the laboratory test. Assuming that

the forces would not alter through the applied structural modification, a modified strain-time history can presumably be simulated from $\{x\}$ and $[He']$. The validity of this assumption is currently also being evaluated.

The purpose being the analysis of the influence of structural dynamics on predicted fatigue, a preliminary analysis of the operating data (strain, vibration) for contributing modes may be in order; this would enable to confine the modification analysis to a limited set of modes. As excitation is normally not measured, ARMA system identification techniques are being investigated for this [21].

3.2 Acoustic Radiation Prediction

The acoustic radiation prediction developments in DYNAMO start from the RAYON software developed by Straco. RAYON is structured as a calculation software that retrieves all definition data (such as geometry, elements, boundary conditions etc.) through a set of proprietary files, as illustrated in Figure 5 [22]. The calculations are based on the BEM technique. The software is been applied with success already for solving various acoustic radiation problems, e.g. for studies on, spacecraft (ARIANE). The main characteristics of this software are currently,

- Element Types. The triangular 3 node element and the quadrilateral 4 node element are fully operational. The triangular 6 node element is developed and in validation; the quadrilateral 8 node element is under study.
- Boundary Conditions. Currently supported are pressure, acceleration (vibrating surfaces) and impedance (absorbing material).
- Acoustic Sources. Currently supported are pressure distribution on the body, body acceleration, simple sources in the acoustic domain as monopoles and plane waves.
- Computed Acoustic Field. Currently supported are pressure, velocity and intensity.

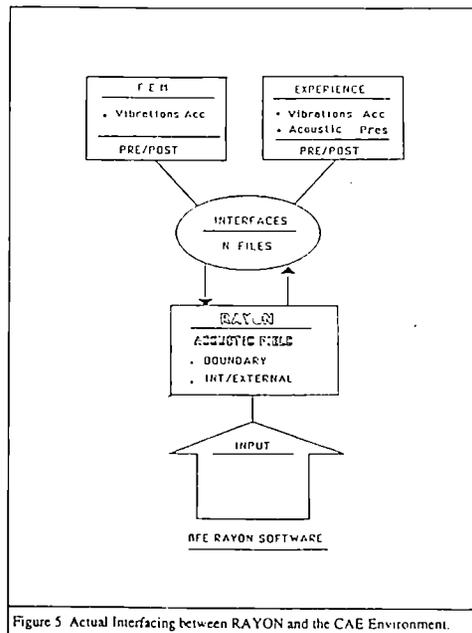


Figure 5 Actual Interfacing between RAYON and the CAE Environment.

The development priorities in DYNAMO are set to address the needs of the automotive industry, specifically,

- The full validation of boundary conditions that are particular to car design, namely vibrating surfaces with absorbing materials.
- Introducing the fluid structure interaction in the calculation. The structure model should be recoverable from a FEM analysis or a CAT prototype test.

To realize these objectives, following improvements were realized during the first project year of DYNAMO,

- Formulation, realisation and verification of interaction between triangular and quadrilateral elements.
- Introduction of boundary conditions with continuous and discontinuous impedance.
- Formulation and realisation of the fluid-structure coupling, allowing the structure to be defined by a structural impedance matrix $[K] - \omega^2 [M]$ (direct coupling), or by a set of modes (modal coupling).

The verification of these developments has necessitated the development of a system that integrates RAYON with the CAE environments at the industrial partners sites, specifically with PDA/Patran and MSC/Nastran in use at the Fiat Research Center. This was realised by using a data exchange mechanism for product design geometry and analysis data that was developed in a former ESPRIT project, no. 322, CAD Interfaces, referred to as the CAD*I neutral file [23]. LMS has developed a CAD*I preprocessor for MSC-Nastran, enabling the exchange of FEM definition data and structural dynamics analysis results [24]. The interface is used to exchange FEM data with LMS CADA-X LINK [25], a product that, for structural dynamics analysis, enables pretest analysis, correlation analysis, design sensitivity and FEM model updating. A joint development between LMS and PDA Engineering, has resulted in a CAD*I preprocessor with similar capabilities for PDA/Patran, PAT/LMS. Through PDA/Patran, structural dynamics analysis results from various analysis codes can then be exported to the CAD*I neutral file. The system that is currently realised is illustrated in Figure 6.

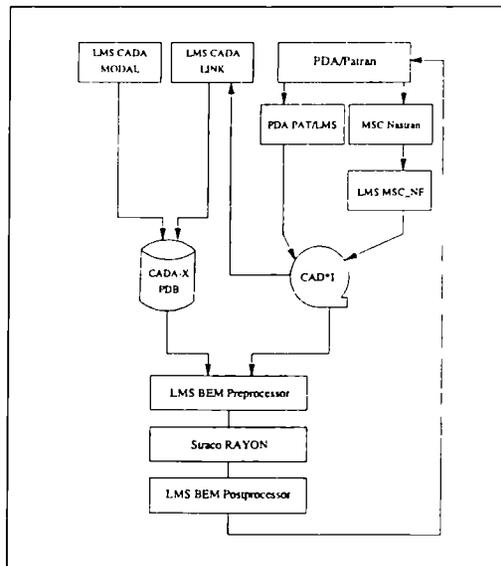


Figure 6 The Integration of RAYON in the CAE Environment at Fiat Research Center

The CAD*I neutral file is created based on the PDA/Patran or MSC/Nastran data base. It is read by a dedicated BEM preprocessor in which following functions, illustrated in Figure 7 are implemented.

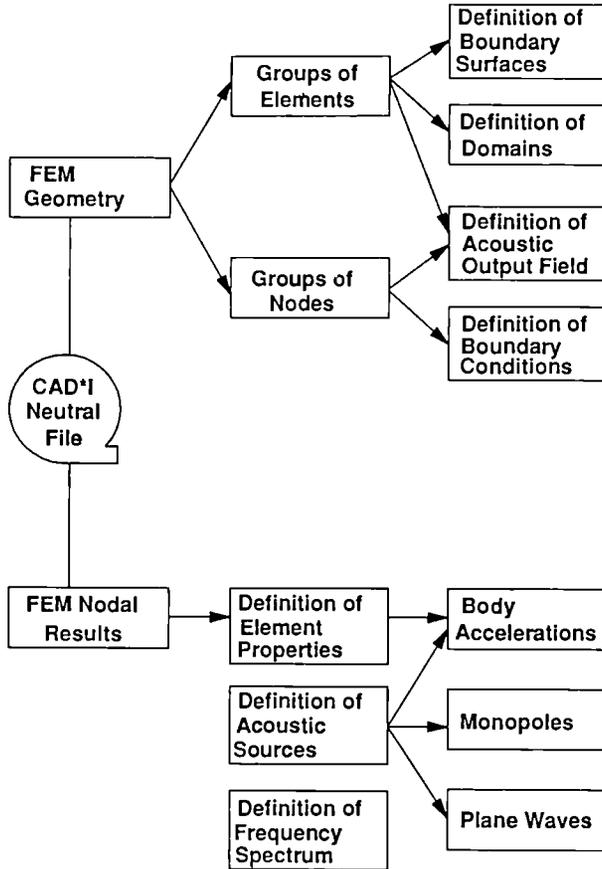


Figure 7 BEM Preprocessor Functions

In the current state of the system, the results of the RAYON calculation are processed for display in PDA/Patran.

Figures 8, 9, 10 illustrate the transition of a simple test case through the different phases of the analysis. The acoustic field produced by the first torsion mode of a plate was taken as a test case.

Figure 8 shows the vibrating plate together with the output mesh in the FEM preprocessor.

Figure 9 shows the transition of the nodal acceleration pattern to element normal accelerations in one of the functions of the BEM preprocessor.

The computed intensity field is shown in Figure 10.

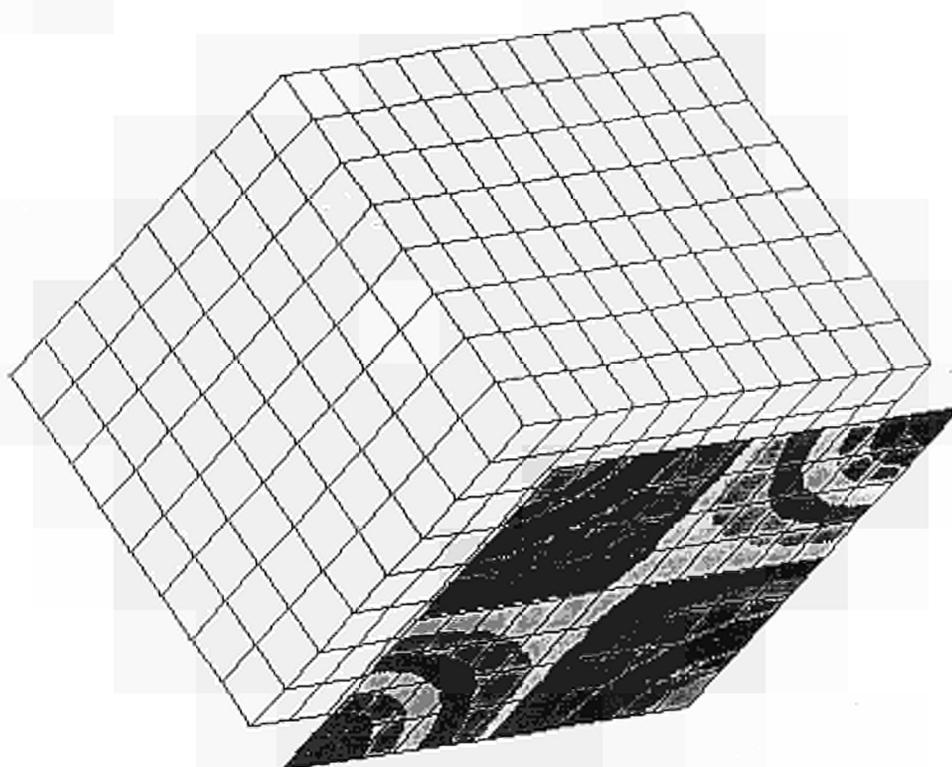
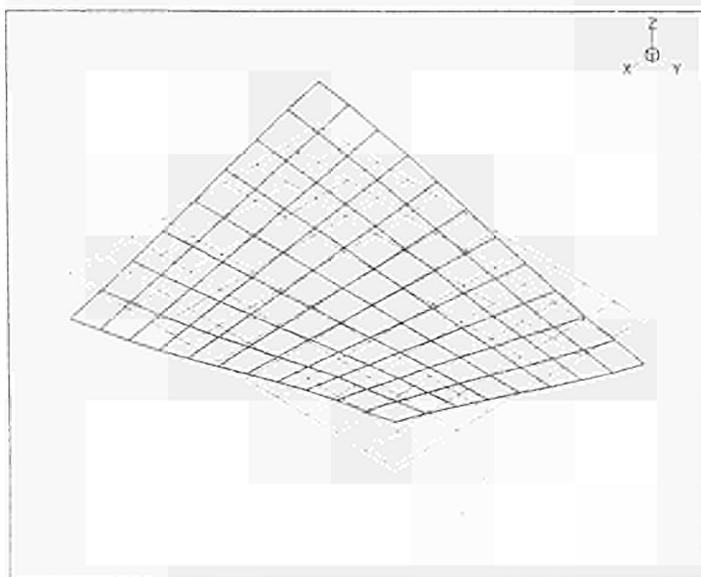


Figure 8 Vibrating Plate and Output Mesh in FEM Preprocessor

LMS CUBI-X



Fmt : Ampl / Phase

Fc
Frequ: 1
334.824

Figure 9A Nodal Acceleration Field (BEM Preprocessor)

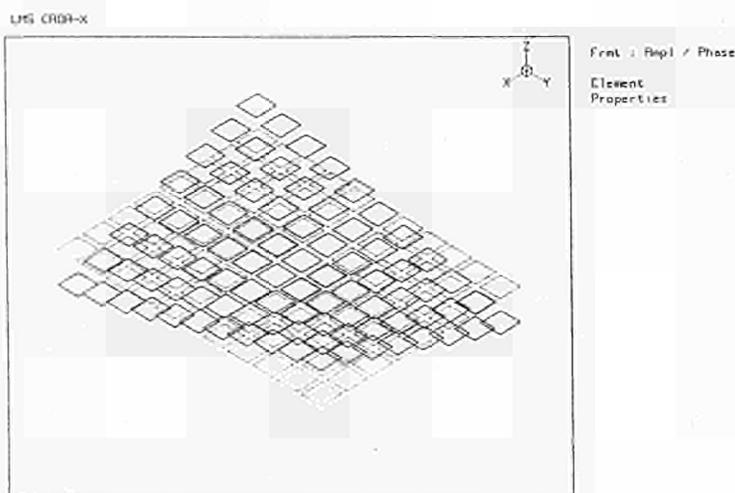


Figure 9B Normal Element Acceleration (BEM Preprocessor)

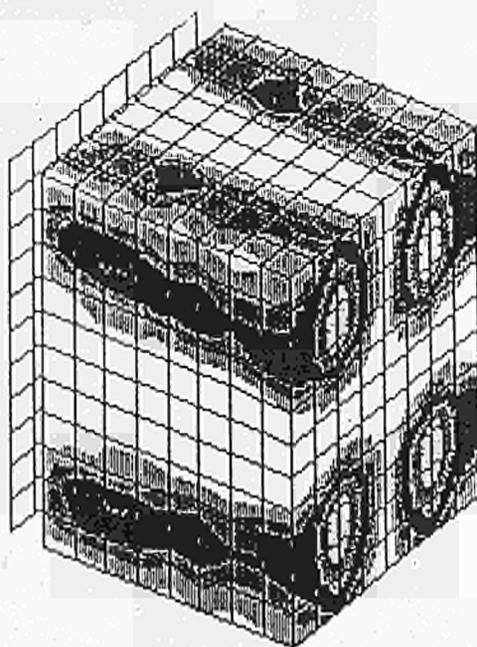


Figure 10 Computed Intensity Field

An extensive series of test cases will be used to verify correctness of acoustic predictions including the fluid structure coupling. This should also learn the limits and correct application of the technique. The test cases focus on requirements of the automotive industry, and range from automotive like panels, over simplified 3D enclosures, to a study on a full vehicle. The approach that is followed for the evaluation on automotive panels consisted off,

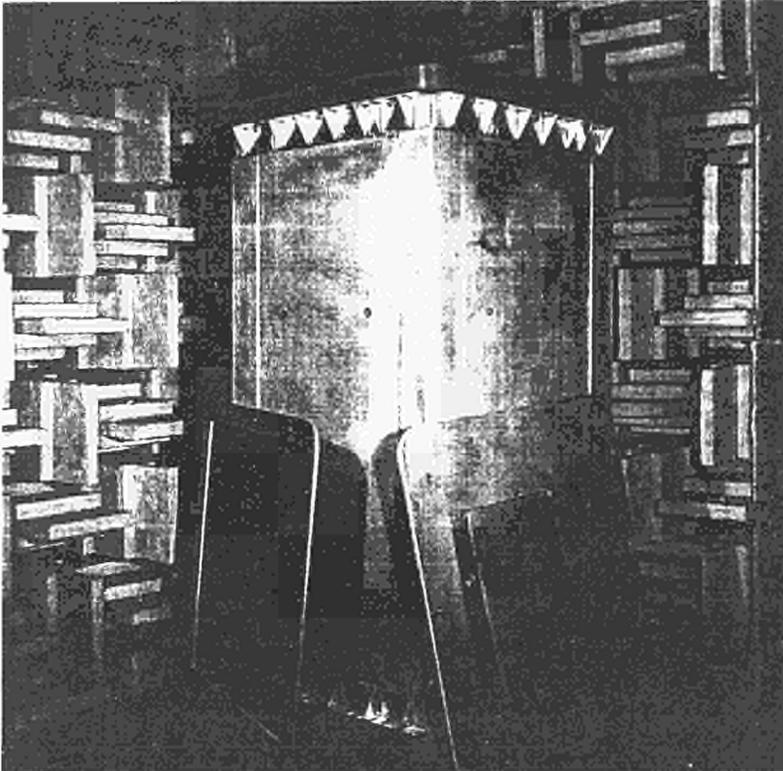
- Superficial distribution of the vibrations in panels induced by external sources; measurements of acceleration and applied forces.

- Measurements of sound pressure levels both in near-field and far-field configurations.
- Generation of data base compatible with the requirements of the BEM prediction code.
- Experimental and FEM analysis of the panels.
- Correlation analysis between FEM and experimental modal analysis and forced response predictions.
- Computation by the BEM method of the acoustic radiation emitted by the vibration panels. Parametric study of the key requirements (e.g. discretization, numerical accuracy) of the computer code.
- Correlation analysis between BEM predictions and the experimental results.

These studies have been applied to plates with various dimpling factors, including the fluid-structure coupling in the analysis process.

A similar approach is being applied for a study on a 3D cavity. This test bed, illustrated in Figure 11, was designed with following goals in mind,

- To check the coherence level between experimental and numerical evaluation with a capability of parametrization on structure and acoustic cavity.
- To understand the influence and way of correct simulation of trimming materials.



Annex 2 - EXTERNAL VIEW OF THE BOX

LAYED TO THE SIDE WALLS THERE ARE THE FRONT VIBRATING
PANELS AND THE INTERNAL DEFLECTING PANEL

Figure 11 External View of the Test Bed. Layed to the Side Walls, there is the Internal Wall.

The test bed is conceived with following specifications,

- In order to begin the experimental and numerical comparison in the simplest configuration: simple shape of cavity, simple shape of walls, decoupling in proper frequencies of the cavity, decoupling among proper frequencies of the cavity walls.
 - In order to study a physical environment quite similar to a car body interior: cavity geometrical dimension similar to car cavity, walls dynamic properties quite equal to car panels.
 - In order to get parametric comparison: capability to change cavity shape and vibrating panels, capability to introduce trimming materials.
- Above specifications have been realized by following concept,
- Parallelepiped shape: 600x600x1500. First cavity frequency at 120 Hz.
 - Wall designed to have only one or two structural modes around the cavity frequencies (90-150Hz).
 - Capability to change both the vibrating panels and the cavity shape.

3.3 Open System Approach

The developments in the DYNAMO project aim at a multidisciplinary analysis capability to solve structural dynamics problems with complex interactions between fatigue, acoustic radiation and vibrations. It will result in a component of a total CAE analysis environment that in almost every design environment is composed of systems from different origin. This has historical reasons, where originally individual software systems were designed for some specific discipline for which they perform best. This 'best-of-class' behaviour of individual software systems is today desirable for best technical capability, but their use in a multidisciplinary environment requires a design that adheres to an open system architecture. The necessity of a common user interface/graphics design in such systems is discussed briefly. The requirement for open data exchange mechanisms was already indicated in the preceding.

User interface technology certainly has been one of the most evolving areas in information technology over the last couple of years. Several standardisation efforts, most of them on stimulus by industry, clearly indicate the need for more user friendly graphical user interfaces, referred to as GUI's [26, 27]. This is especially actual for computer workstations that enable an individual highly interactive user environment. The availability of this hardware has certainly contributed to the wide spread use of CAD and CAE information systems for engineering design.

The systems that execute today on computer workstations mostly use proprietary user interfaces and graphics, putting an increasing burden for understanding how to use a particular system on the design engineer, and even more on the analysis engineer as there is within one company in general more proliferation in CAE systems than there is in CAD systems. This increases unnecessary learning and training periods, and deviates attention from the core of the engineers job: a correct application of the analysis functions that are present in a particular system to conduct the most accurate analysis.

The design of systems around standard GUI's becomes essential if the total CAE environment, that is in use at a particular location, is to be composed of the best components, which in general are not available from one single software vendor. GUI based applications are easier to learn as the user merely has to re-learn specific functionality.

AGUI is composed of following components: a windowing system, an imaging model and an Application Program Interface (API).

The windowing system is a set of tools to build and manipulate windows, draw lines, handle text, etc. Several windowing systems are being put forward for standardization. The X Window System (developed at the Massachusetts Institute of Technology MIT [28] and NeWS (Network Extensible Window System from SUN Microsystems) are candidates. The former has found most wide spread acceptance among computer vendors.

The imaging model defines how graphics are placed on the screen. Examples are Display PostScript, QuickDraw (on the Macintosh), PHIGS, PEX (Phigs on X Windows) [29], GPI of Microsoft's Presentation Manager, etc.

The API defines functions that activate pre-programmed components for developing a user interface, such as menus, scroll-bars, editors, etc, as well as procedures for binding the user interface with the underlying applications. Examples are XUI (of DECWindows), CXI (of HP and Microsoft), the NextStep API.

The OSF/Motif GUI - OSF (Open Software Foundation) is a non-profit organisation established in May 1988 by HP, DEC, Bull, EBM, Nixdorf and Siemens - has a defined windowing system and API as illustrated in Figure 12. The imaging model is currently not established, but it is likely to adhere to PEX.

OSF/Motif is built on top of the X Window System Version 1.1 Release 3.0, and is written in the C language, which makes it a highly portable user interface. In fact, it can run on any workstation which is compatible with the X Window System.

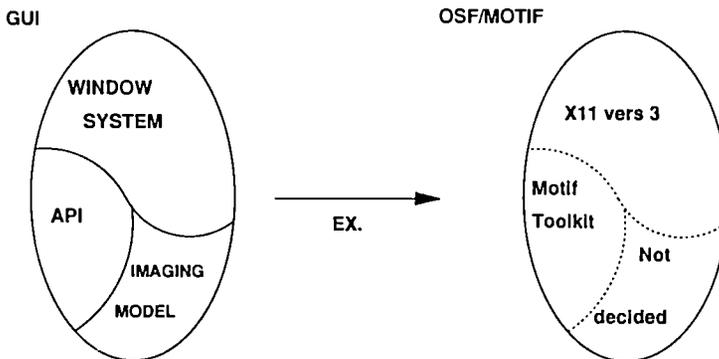


Figure 12 Components of OSF/Motif

The prototype implementations of the DYNAMO system adhere to the OSF/Motif standard. A typical screen layout of an application developed under OSF/Motif is illustrated in Figure 13.

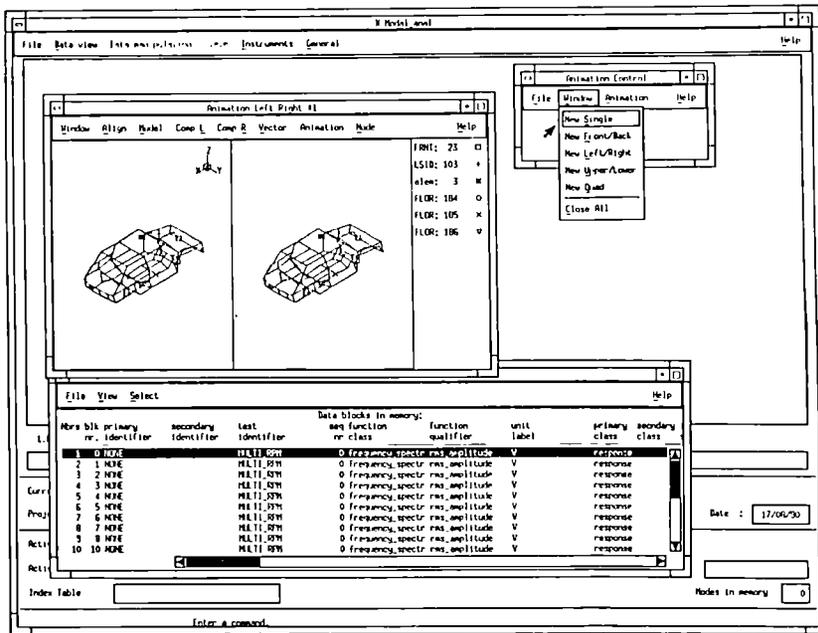


Figure 13 Screen Layout of a CAE Application Developed under OSF/Motif

4. CONCLUSIONS AND OUTLOOK

During the first year of the DYNAMO project, considerable progress was made in each of the addressed technology domains:

- improved crack initiation and crack growth based fatigue analysers,
- extended element types, boundary conditions and acoustic sources for acoustic radiation prediction,
- improved operating vibration data analysis techniques.

At the same time, the basic methodology for integrating structural modelling with both fatigue analysis and noise prediction was spelled out and the development initiated. For fatigue analysis, this mainly involves the modelling and quantification of the effect of resonance induced strains on the life time, enabling the prediction of appropriate design modifications.

For acoustic radiation, this involves the coupling of the structural models with the acoustic BEM models. A prototype integration, including all required interfaces, was realised. An extensive test program, to relate modelling with experimental results for a set of laboratory systems was defined and initiated.

Also, the definition of the required open system architecture, based on a standardised user interface approach (X-windows/OSF-Motif) was completed and the development of the necessary platform building blocks was started.

During the remaining part of the DYNAMO project, the development of the integrated design approach will be completed. For fatigue analysis, the emphasis will be on introducing modal filtering and modal editing tools for solving resonance fatigue problems by structural modifications. The developed methodology will be validated by means of a resonance fatigue failure in a car.

The integration of structural dynamics modelling in the acoustic radiation prediction will also be completed. Properly designed experiments will further be executed to validate the approach. Finally, the full implementation of the developed prototyped approaches on the integration platform will be realised by the end of the project.

Since the developed methodology is general, and not limited to the car industry, some case studies will be worked out for other industrial sectors (aircraft industry, consumer products,.....)

ACKNOWLEDGEMENTS

This paper summarizes the activities of the first year (010589-310490) of the ESPRIT project no. 2486 DYNAMO. The paper reflects contributions of all partners to the project. The authors acknowledge the financial and technical support of the EC and the two project reviewers, Dr. Weibel, InterKeller, Zurich Switzerland, and Dr. Weber, ZF, Friedrichshafen FRG.

REFERENCES

- [1] TOPPER, T.H. and BYRE COWDA C.F.B., "Local stress-strain approach to fatigue analysis and design", ASME Conference "Design Engineering", Chicago, 1940
- [2] PARIS, P.C. and ERDOGAN, F., J.Basic Engineering, 85D, 1963, p.5.
- [3] "Fatigue under complex loading: Analysis and Experiment", SAE Advances in Engineering series, Volume 6, Society of Automotive Engineers, Warrendale, Pa, 1977.
- [4] MORTON, K., MUSIOL, C. and DRAPER, J.M., "Local stress-strain analysis as a practical engineering tool." SEECO Conference "Digital Techniques in Fatigue", City University, London, 1983.
- [5] AUSTEN, I.M., "Quantitative assessment of corrosion fatigue crack growth under variable amplitude loading", Report from commission of European Communities, E.C.S.C. Sponsored Research Project, ECSC Agreement NO:Y21O.KE/810 (F5.5/83), July 1987.
- [6] TIVEY, C.J., "The validation of fatigue life prediction techniques in support of a major vehicle project", Proceedings of the Institute of Mechanical Engineers, Part D, Vol 200 NOD5, 1986.
- [7] CHERTOCK, C., "Sound radiation from vibrating surfaces", J.A.S.A., Vol.36 (7), 1967.
- [8] CHAN, J., SCHWEIKERT, G., "Sound radiation from an arbitrary body", J.A.S.A., Vol.35 (1), 1963.
- [9] COPLEY, L.G., "Fundamental results concerning integral representation in acoustic radiation", J.A.S.A., Vol.44(I), 1968.
- [10] SCHENCK, H.A., "Improved formulation for acoustic radiation problems", J.A.S.A., Vol.44(2), 1968.
- [11] HAMDY, M.A., "Une formulation variationnelle par équations intégrales pour la résolution de l'équation de Helmholtz avec des conditions aux limites mixtes", C.R. Académie Scientifique de Paris, Série II, t.292, pp. 17-20, 1981
- [12] HAMDY, M.A., VILLE, J.M., "Effect of finite length on sound radiation", A.I.A.A. Journal, vol.20, nr. 12, 1982.
- [13] HAMDY, M.A., "Une formulation variationnelle par équations intégrales pour le calcul de champs acoustiques finitaires proches et lointains", Thèse d'Etat, Université de Technologie de Compiègne, 1982.
- [14] VAN DE PONSEELE, P., "3D-Acoustic radiation prediction models based on a numerical solution of the Helmholtz equation", K.U.Leuven, Dept. Werktuigkunde, 9OD4, Mei 1990.

- [15] SAS, P., VAN DE PONSEELE, P., SNOEYS, R., "Prediction of near field intensity problems based on modal deformation patterns", Vehicle Noise and Vibration, I.Mech.E. Conference 1984-1985, pp. 189-195
- [16] MARIEM, J.BEN, "Etude du couplage électroacoustique par une méthode d'éléments finis de surface", Thèse de 3ième cycle, Université de Technology de Compiègne, 1984.
- [17] JEAN, P., "Une méthode variationnelle par équations intégrales pour la résolution des problèmes intérieurs et extérieurs de couplage élastoacoustique", Nouvelle thèse soutenue en juin 1985, cette thèse soutenue en juin 1985, cette thèse a déjà fait l'objet d'une publication.
- [18] WALSH, S.J., LALOR, N., "Optimization of Noise Control Measures in Complex Lightweight Sheetmetal Structures by using Energy Flow Analysis", Brite Technological Days, 1988, Brussels.
- [19] TSANG, N.F., 1990, "Use of Dynamic Strain Measurements for the Modelling of Structures", Proc. of 8th. IMAC, pp. 1246-125 1, Orlando Fl.
- [20] LIEFOOGHE, D., 1990, "Application of Dynamic Analysis Methods in Fatigue Lifetime Calculations", to be presented at the 15th International Seminar on Modal Analysis and Structural Dynamics, Leuven, Leuven Belgium.
- [21] KITAGAWA, G., AKAIKE, H., "On TIMSAC-78", published in Applied Time Series Analysis, IIH', Ed. D. Findlay, Academic Press, New York, pp. 499-547, 1981.
- [22] "Manuel d'Utilisation du Module Rayon Tridimensionnel", Vers. 1, Straco, 1987
- [23] THOMAS, D., J. VAN MAANEN and M. MEAD (Eds.), "Specification for Exchange of Product Analysis Data, Version 3", Research Reports ESPRIT, Project 322, CAD Interfaces (CAD*I), Vol. 2, Springer Verlag, 244 pp.
- [24] "MSC-NF-MS/Nastran CAD*I Neutral File Interface, User and Technical Manual", Rev 2.2, LMS International, 1989.
- [25] "CADA-X Link User Manual", Rev 2.4, LMS International, 1989.
- [26] HAYES, F., "A Guide to GUI's", Byte, July 1989, pp. 250-257
- [27] SOUTHERTON A., "Making Unix easier to use", Unix World, July 1989, pp 62-67
- [28] SOUTHERTON A., "The Story of X", Unix World Networking 1989, pp 69-73
- [29] ROST R., et al., "PEX : A network-transparent 3D Graphics System", IEEE Computer Graphics & Applications, July 1989, pp 14-26

DRIVING ROBOTS VIA NEUTRAL INTERFACES

U.I. KROSZYNSKI, T. SØRENSEN, T.C. CLAUSEN, and E. TROSTMANN
Control Engineering Institute, Technical University of Denmark
Building 424, DK-2800 Lyngby, Denmark
Tel. +45 45934419 Fax +45 42884024

ABSTRACT

Standardization of the product description interface between CAD and manufacturing application systems, as well as of the Explicit Machine Control (EMC) description interface between these and the real-time production equipment, are pre-requisites for the implementation of CIME systems. Standardization will allow to replace software components and production machines by more effective ones as they appear in the market, leading to the application of customized CIME systems based on multiple vendors.

ESPRIT project 2614/5109, NIRO, deals with the definition of a neutral description of the kinematics of mechanisms as part of the product description interface, and with the establishment of an intermediate code for robot explicit control, as part of the EMC interface. It also deals with a neutral, high level programming language for robots.

The goals of the project and results achieved at this stage are outlined. Selected aspects of the EMC description interface for robotics applications are discussed. Examples that illustrate the exchange formats of the interfaces are presented.

An industrial application is described, concerning the welding of modular ship sections. Finally, the influence of the project on standardization efforts is also addressed.

1. INTRODUCTION

A key factor for a successful implementation of Computer Integrated Manufacturing and Engineering (CIME) is the availability of a neutral digital description of the product, in a way that can be accessed and modified by the suppliers as well as the subcontractors and factories involved in making the product. This way, a small firm, specializing in, say, handles for car windows, can draw specifications from different car manufacturers, and propose diverse designs, irrespective of which CAD system they happen to use. The same digital description of a window handle is employed by CAM systems to program the NC machines, by Computer Aided Production Planning (CAPP) systems to plan the production, etc. (1).

The idea of employing neutral digital descriptions for the tasks involved in manufacturing the products, allows to perform the actual production in different factories, with machines and robots of different make. In particular, the application of robots with other production equipment and sensory control, has drawn the attention to the importance of a comprehensive, standardized Explicit Machine Control (EMC) description (2,3,4).

Leaving aside logistic and managerial functions, those two main neutral interfaces can be identified in a Computer Integrated System kernel as sketched in Fig. 1.

Function	Computer Aided Tools	Neutral Interface
Design	CAD / CAE	Product description EMC description
Manufacturing	CAM / CAPP	
Production	NC / CNC / DNC	

Fig. 1 : A CIME system kernel identifying two main interfaces.

The first interface refers to the product description, which should include all the relevant characteristics of the object to be produced. CAD systems usually generate a description of the geometrical characteristics, i.e. shape, dimensions, and topology, of each single component, subassembly, and entire machine part. CAD application modules, or dedicated software packages and systems, supplement the product description with, e.g. the kinematics that governs the moving parts of the product.

Traditionally, a set of engineering drawings and bills of materials, fulfils the role of the product description interface. Drawings convey the information to the manufacturing phase and to other design bodies. Although important as a medium for human control and documentation, drawings are unfit for computerized processing. Digital product descriptions are therefore the natural complement for this interface. Examples of such interfaces are associated to the acronyms IGES, SET, VDAFS, and lately, to the more comprehensive ones PDES, CAD*I, and the international standard STEP (5,6).

The second interface refers to the actual programs that drive the production equipment that makes the product. These can range from instructions to perform a drilling cycle to produce a pattern of holes, to a complex program for assembly of a mechanism with a robot. While EMC descriptions for numerically controlled machines are successfully realized with programming languages like APT, or via cutter location files (CLFILE) generated by CAD application packages, the situation is not quite as mature with robots. Neutral descriptions used for robotics, as for example IRDATA (7) are not yet widespread.

The acute need for standardized interfaces has motivated the international community via the International Standardization Organization (ISO), to elaborate the Standard for the Exchange of Product model data (STEP) (8) for product description on one side, and the Intermediate Code for Robots (ICR) (9) on the other. A standardized high level robot programming language, the Industrial Robot Language (IRL) (10), is also addressed by ISO.

Parallel efforts were launched under ESPRIT. In particular, the Communications Networks for Manufacturing Applications (CNMA), and the CAD Interfaces (CAD*I) projects, directly contributed to the standardization efforts at international level.

ESPRIT project 322, CAD*I (11,12), dealt with the specification and testing of neutral descriptions for geometry models (13), and finite element models (FEM) in CAD / CAE.

Direct offsprings of the CAD*I project are currently underway:

- ESPRIT project 2010, NEUTRABAS, which deals with the extension of STEP work for the shipbuilding sector,
- ESPRIT project 2195, CADEX, which aims at producing STEP processors for commercial CAD systems, and

- ESPRIT project 2614/5109, Neutral Interfaces for Robotics (NIRO), which deals with neutral descriptions of the kinematics of mechanisms and with robot EMC descriptions via ICR and IRL.

In this article we focus on the NIRO project, outline its goals, describe its current status and the results achieved, and attempt to foresee its potential benefit for industrial applications.

The NIRO consortium is composed by robot vendors, industrial enterprises, software houses, and research institutions from five European countries (14,15). It is primarily the industrial partners, FIAT in Italy, CASA in Spain, and Odense Steel Shipyard in Denmark, who will incorporate the project results and serve as demo sites to appraise the impact of driving robots via neutral interfaces for their specific, widely different applications.

2. NEUTRAL INTERFACES FOR ROBOTICS

The situation in robotics resembles the early days of computers, where every type of computer had its own programming language. For instance, computer users used to buy turnkey word processing systems, including hardware and software from one company, just as robots are purchased today. It is today quite simple to move a program from one computer to another, because of standardization in hardware and programming languages. Software houses are specializing in programs, and hardware/firmware manufacturers are specializing in computers. Experience shows that this is a much less expensive way to produce word processing systems. Besides, the consumer is able to choose freely the software/hardware combination which fits him best and change one of the components if necessary.

Robot simulation packages have proven to be a valuable tool for the off-line programming of robots, especially when tasks are to be changed frequently. On such systems, the programmer employs digital models of the robots and the environment, to program, analyze, simulate, and verify tasks, without accessing the actual robots. After iterative trials, when the simulation is satisfactory, i.e. synchronization is achieved, singularities and multiple robot arm configurations are resolved, and paths are free of collisions and time optimized, the system eventually produces programs to be transferred to the robot controllers.

A representative example for the use of such a system is to "mirror" a robot spot welding program in car production, so that two robots, each on its side of the car body, can work simultaneously. Although the task is mirrored, the robots are not, and diverse verification tests need to be done in order to ensure collision free paths on both sides.

Some advanced CAD systems (e.g. CATIA), provide application modules for robotics simulation, which are compatible with the other modules of the system. More usually, however, robot simulation systems are stand-alone software packages. They need therefore the geometric and kinematic models of the robots, objects, and environment, to be created by the operator, before he is able to develop the programs for simulation. Examples of commercial robot simulation systems available on the market are GRASP (16), and ROBCAD (17).

Since the geometric modeling capabilities in a robot simulation system might not be as comprehensive as in a CAD system, this input can be provided via the product description interface by appropriate processor programs. As output, a robot simulation system generates robot programs in some native code. In the case of GRASP, the code is GRDATA. The native code needs to be translated into a particular robot code for each

robot, or, alternatively, into an equivalent neutral code e.g. IRDATA or ICR, assuming it is supported by different robot manufacturers.

A demonstration at the Fifth ESPRIT Technical Week in Brussels in November 1988 showed the successful exchange of (geometry) product data between several CAD systems via the CAD*I interface. However, only one robot simulation system and one robot were tried in that event for the EMC interface (18) using IRDATA.

In the framework of the NIRO project more emphasis is put on this interface. The transfer of kinematic data was selected as an area of interest for the extension of the product description interface. One of the goals is to evaluate the ISO STEP extension proposal part 105 (19), by transferring models between simulation packages (CATIA and BRAVO3 kinematics, KISMET, and GRASP). The other main effort in the NIRO project concerns the EMC description interface for robots. The project aims at testing and proposing modifications and upgrades for IRL, a high level structured language for robot programming, as well as for ISO ICR, a standardized intermediate code at the level of, say, ASSEMBLER instructions. The robots to be employed are REIS, ABB IRB2000, HITACHI PW10, and HIROBO (Fig. 2).

Since processors for the robots in Fig. 2, as well as for the KISMET and GRASP systems were already coded employing IRDATA as neutral description (20,21), a preliminary activity for extrapolating ICR functionality from tests with IRDATA was completed.

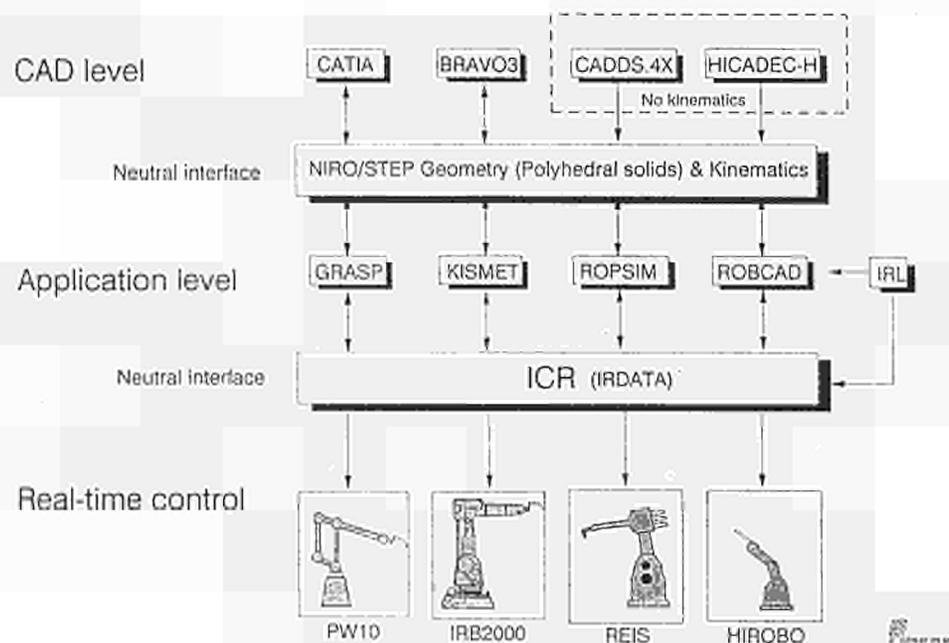


Fig. 2: Schematics of the NIRO implementation goals. The ROBCAD and ROPSIM systems are beyond the project scope.

A neutral control language is not limited to cope with instructions that can be executed on a special robot. It can be constructed to cover a more general instruction set which reflects the generic functionality of robots. This is a significant feature of a neutral interface for robot programs for maintaining the robustness of the CIME architecture.

The activities within NIRO conform to the following pattern:

- Identification of the relevant interfaces,
- Identification of the information representing the functions and data for planning, programming and program execution in robotics, which have to flow across the interfaces,
- Formal specification of the data types and statements, their semantics, and their syntactical representation,
- Development of processor programs, testing and enhancement,
- Demonstration and documentation, and
- Contribution to international standardization activities.

3. CURRENT STATUS - KINEMATICS

One of the first results in NIRO, is a specification of the subset of ISO STEP neutral product description to be used as a reference model (22). This subset comprises a minimum of geometry and topology in the form of polyhedral solids (faceted boundary representations), and supports only some (prismatic and revolute) lower pairs for kinematics. The reason for such a limited range is that polyhedral solids are commonly used for robot modeling in simulation systems, and perhaps more than 90% of the robots can be characterized by means of prismatic and revolute pairs. The underlying philosophy is to work with a relatively simple, yet ample family of models focusing on the kinematics part, and to show that the transfer of such models is operational on the commercial systems available to the project partners (Fig. 2).

Several methods to describe the kinematics of mechanisms are found in the engineering literature. The neutral description for the transfer of robotics models must be able to include these methods and notations, regardless of the method originally used in the sending system. Since the known principles for the description of kinematics can be expressed in terms of homogeneous transformation matrices, it was decided to employ the entity "axis2_placement" (8) as the general placement method.

The actual medium selected for the product interface is in the form of an ASCII file, i.e. a textual mapping of the neutral description. The format complies to strict rules (23). The STEP interface, however, is defined at the more abstract level of data structures, and could be realized as e.g. records in a database accessed via Standard Query Language (SQL) commands instead.

The STEP specification uses EXPRESS (24) as a formal data structure definition language. In Fig. 3(a), the EXPRESS definition of a kinematic joint is given as an example. Its mapping to a file format was defined in the specification phase of NIRO, and is shown in Fig. 3(b). An EXPRESS parser program developed at McDonnell Douglas was applied to check the formal consistency of the NIRO specification.

(a) STEP specification in EXPRESS	(b) NIRO realization on a textual file format
<pre> ENTITY kinematic_joint; first link : kinematic_link; second link : kinematic_link; pair : kinematic_pair; UNIQUE pair; END_ENTITY; </pre>	<pre> kinematic_joint_occurrence = ENTITY_NAME "=" kinematic_joint_keyword "(" ENTITY_NAME "," ENTITY_NAME "," ENTITY_NAME ")" ";" </pre> <p>e.g.:</p> <pre>#46=KINEMATIC_JOINT(#23,#24,#32);</pre> <p>The joint identifier is #46. #23 and #24 are the neutral identifiers of the first and second links associated to the kinematic pair #32.</p>

Fig. 3 : The neutral description of a kinematic joint.

A tabular overview of the entities considered in NIRO-STEP for geometry and kinematics is presented in Fig. 4. Entities marked with an asterisk are terminal ones, i.e. they do not refer to other entities. The indentation illustrates the scoping aspect defined in the NIRO specification. The "index_entry" property is a means to associate user defined names to the entity identifiers.

geometry/topology (polyhedral solids)	kinematics
<pre> faceted_brep cartesian_point* poly_loop face closed_shell solid_instance transformation cartesian_point* direction* [index_entry] </pre> <hr/> <p>in the scope of a faceted_brep the following user defined entities were added:</p> <pre> point_direction_pair render_face </pre> <p>Tool definitions and work parameters such as mass and moments of inertia will also be added as user defined entities.</p>	<pre> kinematic_model [faceted_brep / solid_instance] ground axis2_placement cartesian_point* direction* mechanism kinematic_pair* axis2_placement cartesian_point* direction* pair_placement_structure kinematic_link kinematic_joint kinematic_structure [index_entry] </pre>

Fig. 4 : Tabular overview of the STEP entities specified in NIRO.

An overview of the NIRO-STEP processors that are being coded in the framework of the project is sketched in Fig. 5.

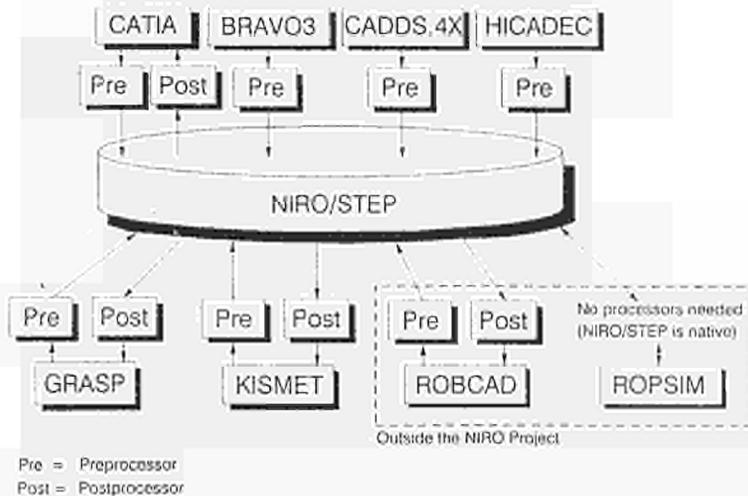
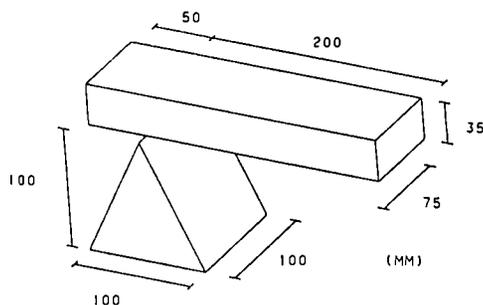


Fig. 5 : Pre- and Post-processor programs to be coded within NIRO.

A complete example of a NIRO-STEP file representing a simple model is presented in Fig. 6.

```
STEP;
HEADER;
FILE_IDENTIFIER('esprit test robot',
'1991-05-10T14:30:46',
['T.S., U.K., E.T. and T.G.C.'],
['IS - Control Engineering Institute',
'Technical University of Denmark - DTH',
'Building 424', 'DK-2800 Lyngby - DENMARK'],
'NIRO-STEP L01V03', 'Preprocessor CATNIRO v1.0',
'CATIA (AIX) version 2.3');
FILE_DESCRIPTION(('Simple one DOF rev. robot.',
'Implementation level NIRO spec. L01V03');
ENDSEC;
DATA;
/* polyhedron */
#1 = &SCOPE
#2 = CRTPNT(+0.00E+00, +0.00E+00, +0.00E+00);
#3 = CRTPNT(+5.00E-02, +0.00E+00, +1.00E-01);
#4 = CRTPNT(+1.00E-01, +0.00E+00, +0.00E+00);
#5 = CRTPNT(+0.00E+00, +1.00E-01, +0.00E+00);
#6 = CRTPNT(+5.00E-02, +1.00E-01, +1.00E-01);
#7 = CRTPNT(+1.00E-01, +1.00E-01, +0.00E+00);
#8 = PLYLOP((#2, #3, #6, #5));
#9 = PLYLOP((#3, #4, #7, #6));
#10 = PLYLOP((#4, #2, #5, #7));
#11 = PLYLOP((#2, #4, #3));
#12 = PLYLOP((#5, #6, #7));
#13 = FACE$(, #8, $); #14 = FACE$(, #9, $);
#15 = FACE$(, #10, $); #16 = FACE$(, #11, $);
#17 = FACE$(, #12, $);
#18 = CLSSHL((#13, #14, #15, #16, #17));
ENDSCOPE
FCTBRP(#18, $);

/* polyhedron */
#19 = &SCOPE
#20 = CRTPNT(+0.00E+00, +0.00E+00, +0.00E+00);
#21 = CRTPNT(+2.50E-01, +0.00E+00, +0.00E+00);
#22 = CRTPNT(+2.50E-01, +7.50E-02, +0.00E+00);
#23 = CRTPNT(+0.00E+00, +7.50E-02, +0.00E+00);
#24 = CRTPNT(+0.00E+00, +0.00E+00, +3.50E-02);
#25 = CRTPNT(+2.50E-01, +0.00E+00, +3.50E-02);
#26 = CRTPNT(+2.50E-01, +7.50E-02, +3.50E-02);
#27 = CRTPNT(+0.00E+00, +7.50E-02, +3.50E-02);
#28 = PLYLOP((#20, #21, #25, #24));
#29 = PLYLOP((#21, #22, #26, #25));
#30 = PLYLOP((#22, #23, #27, #26));
#31 = PLYLOP((#23, #20, #24, #27));
#32 = PLYLOP((#20, #23, #22, #21));
#33 = PLYLOP((#24, #25, #26, #27));
#34 = FACE$(, #28, $); #35 = FACE$(, #29, $);
#36 = FACE$(, #30, $); #37 = FACE$(, #31, $);
#38 = FACE$(, #32, $); #39 = FACE$(, #33, $);
#40 = CLSSHL((#34, #35, #36, #37, #38, #39));
ENDSCOPE
FCTBRP(#40, $);
```



```

/* solid instance */
#41 = &SCOPE
#42 = &SCOPE
#43 = DIRCTN(+1.00E+00, +0.00E+00, +0.00E+00);
/* #44 = DIRCTN(+0.00E+00, +1.00E+00, +0.00E+00); */
#45 = DIRCTN(+0.00E+00, +0.00E+00, +1.00E+00);
#46 = CRTPNT(+0.00E+00, +0.00E+00, +0.00E+00);
ENDSCOPE
TRNSFR(#43,$,#45,#46,$);
ENDSCOPE
SOLID_INSTANCE(#1,#42);

/* solid instance */
#47 = &SCOPE
#48 = &SCOPE
#49 = DIRCTN(+1.00E+00, +0.00E+00, +0.00E+00);
/* #50 = DIRCTN(+0.00E+00, +1.00E+00, +0.00E+00); */
#51 = DIRCTN(+0.00E+00, +0.00E+00, +1.00E+00);
#52 = CRTPNT(+0.00E+00, +1.25E-02, +1.00E-01);
ENDSCOPE
TRNSFR(#49,$,#51,#52,$);
ENDSCOPE
SOLID_INSTANCE(#19,#48);

/* kinematic model */
#53 = &SCOPE
/* mechanism */
#54 = &SCOPE
/* base frame */
#55 = &SCOPE
#56 = DIRCTN(+1.00E+00, +0.00E+00, +0.00E+00);
/* #57 = DIRCTN(+0.00E+00, +1.00E+00, +0.00E+00); */
#58 = DIRCTN(+0.00E+00, +0.00E+00, +1.00E+00);
#59 = CRTPNT(+0.00E+00, +0.00E+00, +0.00E+00);
ENDSCOPE
AXSPLZ(#59,#58,#56);

#60 = REVOLUTE_PAIR(+0.00E+00,-7.85E-01, +8.73E-01);

#61 = &SCOPE
#62 = DIRCTN(+0.00E+00, +0.00E+00, +1.00E+00);
/* #63 = DIRCTN(+1.00E+00, +0.00E+00, +0.00E+00); */
#64 = DIRCTN(+0.00E+00, +1.00E+00, +0.00E+00);
#65 = CRTPNT(+5.00E-02, +0.00E+00, +1.00E-01);
ENDSCOPE
AXSPLZ(#65,#64,#62);

#66 = PAIR_PLACEMENT_STRUCTURE(#60,#61);

#67 = &SCOPE
#68 = DIRCTN(+0.00E+00, +0.00E+00, +1.00E+00);
/* #69 = DIRCTN(+1.00E+00, +0.00E+00, +0.00E+00); */
#70 = DIRCTN(+0.00E+00, +1.00E+00, +0.00E+00);
#71 = CRTPNT(+5.00E-02, +0.00E+00, +1.00E-01);
ENDSCOPE
AXSPLZ(#71,#70,#68);

#72 = PAIR_PLACEMENT_STRUCTURE(#60,#67);

#73 = KINEMATIC_LINK((#66),$,(#41));
#74 = KINEMATIC_LINK((#72),$,(#47));

#75 = KINEMATIC_JOINT(#73,#74,#60);
#76 = KINEMATIC_STRUCTURE((#75),$);
ENDSCOPE
MECHANISM(#76,#73,#55);

/* the ground */
#77 = &SCOPE
#78 = &SCOPE
#79 = DIRCTN(+1.00E+00, +0.00E+00, +0.00E+00);
/* #80 = DIRCTN(+0.00E+00, +1.00E+00, +0.00E+00); */
#81 = DIRCTN(+0.00E+00, +0.00E+00, +1.00E+00);
#82 = CRTPNT(+0.00E+00, +0.00E+00, +0.00E+00);
ENDSCOPE
AXSPLZ(#82,#81,#79);
ENDSCOPE
GROUND(#78,(#41));

ENDSCOPE
KINEMATIC_MODEL(#77,(#54),$,$);
ENDSEC;
ENDSTEP;

```

Fig. 6 : An example of a NIRO-STEP file for a simple model.

The model was designed with the CATIA system v3.2 (under AIX, on an IBM series 6000, RS320 RISC machine), and preprocessed to the neutral format above with a pilot preprocessor coded at the Control Engineering Institute (IfS) of the Technical University of Denmark (DTH). Examples involving a closed mechanism chain were also modelled, and will be employed in the model exchange test suites. The "pretty print" NIRO-STEP files are too large to be included in this article. However large, the processing time is only a few seconds on the RISC machine. Post-processors to recover the models in the different systems are currently being developed. The ROPSIM system (25) (see Fig. 2), developed at the IfS, DTH, extends the NIRO-STEP schema to deal with dynamics simulation in robotics.

4. CURRENT STATUS - INDUSTRIAL ROBOT LANGUAGE

IRL is an international standardization initiative organized by ISO. IRL is at the moment at the proposal level, and is being tested within the NIRO project. Project partners are reviewing the proposal and are designing a compiler from IRL to the robot control interface.

IRL is an explicit robot programming language. Thus, every single "move" operation requires one statement. The IRL definition is based on concepts of modern structured programming. Therefore, it includes the elements for modularization, program flow control, procedures, functions, local and global constants and variables, block concept, recursivity, etc. In a future extension IRL will also provide for parallel execution of program blocks.

The original definition suffered from many inconsistencies and syntax errors. The major problem was that it merged every proposal from industry, mixing e.g. PASCAL, C, BASIC, ASSEMBLER, BAPS, and SRCL. It was therefore necessary to re-define IRL to obtain a useful proposal. The problem was to define a language which could be accepted by industry. It was decided to base it on PASCAL, a commonly used programming language. A new definition was proposed early in 1991 and the NIRO work is now based on this proposal.

As a result, a compiler from IRL to ICR has been designed. The first version of the compiler supports the most basic elements of robot programming, especially motion. Data manipulation will be considered in later versions.

5. CURRENT STATUS - ROBOT CONTROL

Most high level robot languages build on the same basic philosophy: They shall be easy to use by humans. The structure and instruction set of languages like AML and ARLA are designed so that they are easy to program with, easy to understand, etc.

However, off-line programming by means of computerized tools poses other demands for a robot language. It is no longer a need to make it easy for humans to read, since programs are to be exchanged between machines. Thus, the language has to be optimized with respect to other demands, and must be:

- Generally usable. It is important that the off-line programming language consists of a broad instruction set. Instructions for both general data manipulation and robot control are needed.
- Easy to generate. The language shall be easy to generate for programming systems. It shall also be easy to combine program parts from different programming systems (linking).

- Easy to read and execute for machines. An off-line programming language shall reflect that computers execute many simple instructions faster than a few complex ones.

These demands are met by ISO's draft proposal to a robot code interface. The draft proposal is called Intermediate Code for Robotics, ICR.

An intermediate language is a kind of universal, machine independent, Assembler language. It is placed at the level between machine code and high level languages like PASCAL and C. As a consequence, intermediate languages do not include high level instructions like e.g. $C = A^2 + 3 * B$. Still, it is possible to construct these out of a combination of more basic orders.

The most important quality in using the intermediate level for off-line programming is that all higher languages can be mapped onto the same intermediate code. The intermediate code is, therefore, suitable as a neutral interface.

It is also very important to select a suitable instruction set for the robot code. Much programming effort focused on motion control, but other tasks need to be included. For example arithmetic operations, database integration, and error recovery constitute a major fraction of a control program. These entities can be denoted data "manipulation" instructions. Sometimes only ten percent of a program's code concerns motion directly.

The instructions for physical robot control (motion, I/O, etc.) have been selected on the basis of experience gained from other robot languages. This has led to a fundamental robot control instruction set, although more sophisticated commands (e.g. sensor control) are still missing. The data manipulation instructions of the code have been designed following the principles of low level languages for computers.

ICR is based mainly on IRDATA, and the basic concepts and main elements of both codes are similar. Since the initial ICR proposal was not stable, the first tests of the robot control interface were based on IRDATA. These tests were done in order to examine the possibilities of reusing parts of an ICR interpreter and the feasibility of using the intermediate level for robot control.

Prototype versions of the ICR software are now installed at the project partners sites. Two different approaches in the transfer of an ICR robot program to the robot are used. One approach is to translate ICR into the native robot language, and other is based on the use of an ICR interpreter, as it was carried out with the first tests based on IRDATA.

The compiler approach can be used without communicating with the robot controller during program execution, since a native robot program is downloaded to the robot controller. The translator solution has been selected for the Odense Steel Shipyard (OSS) demonstration. The HIROBO robots at OSS are very simple ones without on-line control facilities. The robots are strictly used for welding (i.e. the robots are only doing motion). The ICR to HIROBO code translator therefore only supports the motion commands of ICR.

Alternatively, an interpreter is a piece of software that is able to read a program and to execute what is specified in the program in a step-wise manner. When using an interpreter for executing an ICR program most commands will be handled by the general purpose part of the interpreter. Examples are arithmetical operations and program flow commands. However, commands involving physical actions (e.g. moving the robot, controlling the gripper, and addressing I/O ports) are sent to the robot specific module where they are transformed to the format required by the controller and sent on-line into it. ICR interpreters for the ABB IRB 60 and IRB 2000 robots have been developed. In the CASA demonstration an interpreter is used for controlling a sophisticated on-line sensor correction of a filament winding process performed by a robot.

A discussion about the relative advantages of compilers and interpreters in robotics can be found in (21).

Fig. 7 shows an example of ICR code to move a robot continuously between two positions. The code is shown in its "symbolic" format. A "number" format also exists, which is faster for machine execution.

```

1,PBEG,"TEST";
2,BLKBEQ,0,1;
3,DECLVAR,0,"POS_A",ROBTARGET,0,"POS_B",ROBTARGET;
4,PUSHT,#(ROBTARGET,#(POSE,#(POSITION,X,Y,Z),ORIENTATION,a,b,c,d));
5,POPT,"POS_A";
6,PUSHT,#(ROBTARGET,#(POSE,#(POSITION,X,Y,Z),ORIENTATION,a,b,c,d));
7,POPT,"POS_B";
8,PUSHT,"POS_A";
9,LMOVE,1,W;
10,PUSHT,"POS_B";
11,LMOVE,1,W;
12,GOTO,8;
13,BLKEND;
14,PEND;

```

Fig. 7: An ICR Program.

Line 1 indicates the beginning of the program TEST. In line 2 a block is reserved for the program. In line 3 space is reserved for the two target positions (robtargets) POS_A and POS_B. A "robtarget" consists of a position and an orientation (sometimes also information about external axes). In line 4 the first robtarget is pushed on top of stack. In line 5, the robtarget on top of stack is popped and stored in the variable POS_A. Lines 6 and 7 are as 4 and 5 for POS_B. Line 8 is used to place the robtarget in POS_A on top of stack. In line 9 the robot is instructed to move to the robtarget on top of stack. Lines 10 and 11 are as 8 and 9. Line 12 is an instruction to go to line 8 and start again from there. Lines 13 and 14 terminate the block and the program.

The review of the ICR specification is nearly finished, and efforts are now shifting towards enhancements and implementations. The tests and reviews have shown minor weaknesses within the present scope of ICR. Examples are poor capabilities to handle external axes and an inexpedient point-to-point move command. The tests also showed the need to extend the scope of ICR to cope with new applications. The work planned on ICR will focus on these enhancement areas.

Concerning general sensor instructions, the technology break-through in robotics resulting from sensors, has introduced more complexity in robot programming. The main problems are connection of external sensors (receiving data, scaling, etc.) and processing of sensor feed-back within the robot controller. Present controllers are limited to use only a few specific sensor types and are not sufficiently prepared for the integration of different external sensor types. This may lead to the definition of a number of *generic* sensors in the ICR interface. By pre-programmed generic sensor instructions in the controller, a fundamental influence of sensors upon trajectory generation can be

supported, such as in motion control. The NIRO project will propose the definition of such a general sensor instruction set and include it in ICR.

A standardized general purpose sensor instruction set has major advantages in robot programming. However, some applications are better dealt with the use of application dedicated sensor instructions. For example, it may be possible to use standard sensor instructions for welding sensors, like laser scanners and light-bow sensors. But it might be more beneficial to use a dedicated sensor instruction set, which is able to fully realize the capabilities of these application sensors. The NIRO project will take as an example a dedicated sensor instruction set for welding.

Robots are normally working together with a number of other machines. The overall synchronization is typically handled by a Programmable Logic Controller (PLC) external to the robot. In smaller working cells it may be beneficial to use the robot for this synchronization, so the total system is simplified. As a NIRO result, the PLC functionality will be included in ICR and implemented on a REIS robot.

6. INDUSTRIAL APPLICATION

From the three NIRO industrial partners sites in Europe, we have chosen to describe the implementation of neutral interfaces in CIME for robot applications in shipbuilding at the Odense Steel Shipyard (OSS) in Denmark.

OSS employs welding robots for the production of modular components, e.g. ladders, for ships. It also employs a number of small, sturdy HIROBO welding robots (about 45 kg. weight), some of which are mounted on gantries, for welding flat ship sections. The modeling of the ship section geometry and the weld paths is performed on the HICADEC-H CAD system for ship design. A dedicated processor uses an IGES description of the model to generate the robot programs. OSS has decided to automate the production of modular flat ship sections. OSS joined the NIRO consortium since it aims at the development of a CIME architecture that is robust enough to accept different CAD and programming systems, as well as robots (controllers).

12 m wide gantries with three HIROBO robots each, can weld sections up to a length of 20 m. Entire sections are then conveyed to larger mounting sites, where they are coupled with other sections. Piping and other appliances are then added to finished modules of up to 800 Tons, which are then placed in position by a giant crane at the dock where the vessel is being built. OSS can handle a very large vessel, or more lesser ones simultaneously, with a combined capacity ranging up to 800.000 TDW.

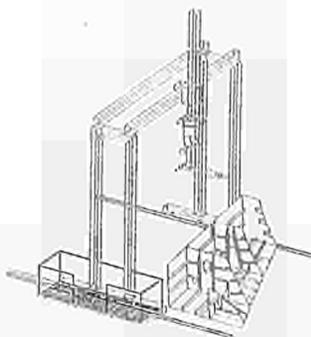
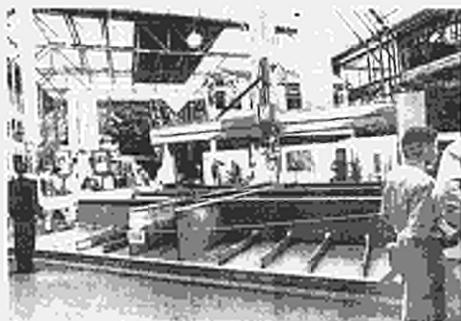


Fig. 8 : Ship section welding at OSS, reproduced from NIRO Status Report (26)

In the production of flat ship sections, an operator places the gantry on a three-dealt parallel compartment along the section. The robots find their way to corners and control deviations from the nominal linear paths via touch sensors. The gantry advances at the rate governed by the slowest of the three robots. A typical section is illustrated in Fig. 8, where only one robot is shown.

The transfer of design data from the modeling phase to robot programming is done via the IGES interface. The automation process is possible at this stage due to the modularity of the sections and the relative simplicity of the welding paths. This is most advantageous for ship hulls with a long, rectangular central portion between the prow and poop sections.

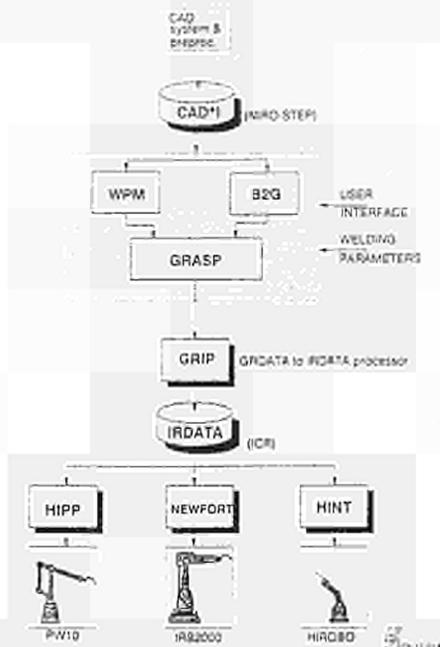


Fig. 9 : Sketch of the IFS demonstration at OSS in January 1991.

In order to make the programming more flexible, standard interfaces are to be introduced:

- between the HICADEC-H system and the robot simulation systems, and
- between the robot simulation systems and the robots.

In January 1991, the application of the latter interface was demonstrated at OSS for the first time, driving different robots to perform the same task (Fig. 9). Since the NIRO-ICR specification was not yet complete, and ICR processors were therefore not applicable, IRDATA was used instead.

7. INFLUENCE ON STANDARDIZATION

Similarly to its predecessor CAD*1, the NIRO project is expected to contribute significantly to the national and international efforts in standardization in the field of robotics. The project has a unique opportunity to test proposed specifications (viz. STEP part 105 and ICR) and assess their applicability by actually implementing them

and trying them out on industrial CIME environments. The feedback from such tests at prototype level is invaluable for the standardization bodies, and result in more consistent, stable, and complete standards and guide-lines for implementation.

NIRO partners also participate in national standardization efforts in their respective countries, and at the ISO subcommittees, where they may influence decisions in the light of project results, for the benefit of industry in general.

8. CONCLUDING REMARKS

In this article, we presented a general overview of the ideas behind the ESPRIT project 2614/5109, Neutral Interfaces for Robotics, NIRO. The current state of the project, and results after the year and a half since it started, are summarized.

The application areas at the level of product description interface, comprise primarily kinematics, and to a lesser extent, geometry. An example of a neutral file for a simple mechanism is given.

The application areas at the level of EMC description interface, focus on a standardized intermediate code for robots, which is to be implemented by the robot manufacturer REIS. The project also addresses a standardized high level robot programming language.

The strategic importance of the project is highlighted in two aspects:

- its influence on currently on-going standardization efforts, and
- its potential impact on industrial automation.

The latter aspect is illustrated by the application of the interfaces to drive welding robots in the production of flat ship sections.

The use of standard interfaces allows to establish CIME implementations based upon multiple vendor software systems and production equipment. This results in a more rational use of design, manufacturing, and production resources, and in the end, in more reliable and better built products.

REFERENCES

- (1) KROSZYNSKI, U. et al. (1986). Standard Interfaces for CAD Data Exchange. Proc. CAPE'86, Copenhagen, Denmark, pp.537-549.
- (2) PALSTRM, B. et al. (1988). CAD Data Transfer to Robot Programming and Control - Application of the CAD*I Neutral File. Proc. 4th. CIM-Europe Conf., Madrid, Spain, pp.139-158.
- (3) PALSTRM, B. et al. (1989). CAD Data Transfer to Robot Programming and Control: Application of the CAD*I Neutral File. The CAD/CAM Yearbook 1989/90. Official Yearbook of the British CAD/CAM Assoc., pp.363-368.
- (4) TROSTMANN, E. (1991). Intelligent Interfaces in Robotics. (Submitted to 1st. Euro-CIM Conference, Mechatronics and Robotics, Aachen, Germany).
- (5) TROSTMANN, E. et al. (1988). CAD Data Exchange via Neutral Interfaces. Proc. ENE'88, SME Technical Paper MS88-376, Baltimore, U.S.A., pp.2.95-2.108.
- (6) KROSZYNSKI, U. et al. (1989). Geometric Data Transfer Between CAD Systems. IEEE, J. of Computer Graphics and Applications, Vol.9, Nr.5, pp.57-71.
- (7) DIN 66313 (1989). IRDATA - Interface Between Programming System and Robot Control. Beuth Verlag.
- (8) ISO (1989). External Representation of Product Definition Data (STEP), ISO/DP 103030.

- (9) ISO (1991). Manipulating Industrial Robots - Intermediate Code for Robots (ICR). ISO/DP 10562-1.2, 75pp.
- (10) Industrial Robot Language (IRL) - Language Description (Draft). NAM in the DIN committee working paper N 96.2.2/15-89, 1989.
- (11) CAD Data Transfer for Solid Models. (E.G. Schlechtendahl, ed.), ESPRIT Research Reports, Vol.3, CAD Interfaces (CAD*I), Springer Verlag, Heidelberg, ISBN 3-540-51826-6, 1989.
- (12) KROSYNSKI, U. et al. (1989). CAD Data Exchange Between Solid Modellers: CAD*I Concluded. Proc. CAPE'89, Tokyo, Japan, pp.363-368.
- (13) Specification of a CAD*I Neutral File for CAD Geometry: Version 3.3. (E.G. SCHLECHTENDAHN, ed.), ESPRIT project Nr. 322, CAD Interfaces (CAD*I), Vol.1, 3rd. ed., Springer Verlag, Heidelberg, ISBN 3-540-50392-7, 1988.
- (14) TROSTMANN, E. (1990). Robotics Research within ESPRIT. Proc. 21st. Int. Symposium on Industrial Robots (ISIR), Copenhagen, Denmark, pp. 377-382.
- (15) BEY, I. (1990). Neutral Interfaces for Robotics. Goals and First Results of ESPRIT Project 2614/5109: "NIRO". Proc. Annual ESPRIT Conf., Brussels, Belgium, pp.559-567.
- (16) GRASP Reference Manual Version 8. B.Y.G. Systems Ltd., UK, 156 pp.,1990.
- (17) ROBCAD Reference manual, Version 2.2. Technomatix Europe N.V., Belgium, 900pp., 1990.
- (18) KROSYNSKI, U. et al. (1989). Neutral Interfaces That Work: Application for Robot Welding. Proc. 6th. IFAC/IFIP/IFORS/IMACS Symposium on Information. Control Problems in Manufacturing Technology, INCOM'89, Madrid, Spain, Preprints Vol.2, pp.779-783.
- (19) Industrial Automation Systems - Exchange of Product Model Data - (STEP) Part 105: Kinematics Information Model - Committee Draft. (ISO 10303 - St. Louis, 1990).
- (20) TROSTMANN, E. et al. (1990). Implementation of an Off-line Programming System Based on the IRDATA Neutral Interface. Proc. 21st. Int. Symposium on Industrial Robots (ISIR), Copenhagen, Denmark, pp. 271-278.
- (21) TROSTMANN, E. et al. (1991). Integration Using the Neutral Interface, IRDATA/ICR, in Robot Programming Does Work!. (Submitted to CAPE'91, Bordeaux, France).
- (22) KROSYNSKI, U. et al. (1991). NIRO-STEP Specification for the Transfer of Robotic Models, Version: L01V03 "February 91" Revised after the Copenhagen Meeting 91.03.11. IfS Rapport Nr. S91.25, 45pp.
- (23) Industrial Automation Systems - Exchange of Product Model Data - (STEP) Part 21, Version 24 July 1990.
- (24) Industrial Automation Systems - Exchange of Product Model Data - (STEP) Part 11: The EXPRESS Language. ISO/TC184/SC4/WG1/N496.
- (25) TROSTMANN, E. et al. (1991). ROPSIM, a Robot Off-line Programming and Real-time Simulation System Including Dynamics. (Submitted to SYROCO'91, Vienna, Austria).
- (26) ESPRIT Project 2614, NIRO, Status Report 1, Kernforschungszentrum Karlsruhe, pp. 90-95, January 1991.

METHODS FOR ADVANCED GROUP TECHNOLOGY INTEGRATED WITH CAD/CAM

DIRK DE MOOR
WTCM/CRIF MACHINEBOUW
CELESTIJNENLAAN 300C
B-3001 HEVERLEE
Tel: + 32-(0)16-286611
Fax: + 32-(0)16-237678
Telex: 25393

E-mail: luk_van_den_noortgate@eurokom.ie

SUMMARY

Group Technology (GT) is the concept of identifying and bringing together similar parts in order to obtain advantages from their similarity for designing and manufacturing purposes. To put Group Technology into practice a coding and classification system is used. Although Group Technology as a concept is considered to be most valuable, a lot of companies have encountered serious problems trying to implement it. Efforts to lower the barriers to Group Technology have resulted in the ESPRIT project 2613 MAGIC. The key development in this project is the elaboration of a Computer Automated Group Technology (CAGT) software. By automating the input from CAD, consisting of features and attributes, human interpretation of engineering drawings during the coding process is excluded. A performing retrieval tool will automate the retrieval of existing similar parts, drawings and manufacturing sequences based on the internal GT representation or on a rough sketch on the CAD system. The internal representation stored in a relational database is a successful start to an engineering database with links to process planning, material requirements planning, manufacturing, purchasing, ...

1. THE MAGIC PROJECT

Objectives

The MAGIC project aims at improving and facilitating the use of Group Technology methods and tools in small and medium sized factories, by automating the methods and by integrating the tools in existing CAD/CAM systems.

The final result will be a Computer Automated Group Technology software including :

- Automatic creation, during the design task on the CAD system, of a computer internal representation of assemblies, subassemblies and parts, including geometrical, functional and production characteristics.
- Automatic retrieval of existing similar parts and drawings based on a rough sketch of the new part on the CAD system or based on user friendly input menus; automatic retrieval of existing assemblies and subassemblies based on functional and morphological characteristics and automatic retrieval of appropriate manufacturing information sequences.

Partners

Industrial partners as well as research centres and software houses are involved in the project MAGIC.

WTCM is the prime contractor of the project. WTCM is a research centre of the Belgian metalworking industry. The department Mechanical Engineering has a research team of 30 engineers, working in the fields of CAD/CAM and FMS.

CETIM is a Technical Centre of the Mechanical Industry in France. The Production Engineering department (40 people) coordinates the research in company management, process planning, machining techniques and Group Technology.

N.V. MICHEL VAN DE WIELE is a medium sized Belgian company (800 employees) manufacturing weaving machines. It has a manufacturing experience of more than 100 years in textile machinery engineering for carpet and velvet weaving machines.

LVD, a Belgium machine tool manufacturer has been involved in mechanical engineering for more than 30 years. The extensive production program covers not only standard machine tools but also special manufacturing systems.

C.M. MARES is an injection mouldmaker for plastics materials since 1952. Almost 90% of their turnover is directed to the automotive industry. The company is specialized in big tools until 40 tons, with complex surfaces.

Eigner & Partner are fast growing German system consultants for the solution of problems in the technical field of CAD/CAM.

CAP SESA INDUSTRIE is a French software house that works on a very large range of different fields of Computer Integrated Manufacturing, covering CAD/CAM, Flexible Manufacturing Systems, networks, data management systems,...

Project Status

ESPRIT-2623-MAGIC is a three year project, presently at the end of its third year. In the first year of the project, two prototypes have been realised: one for the automatic generation of the internal GT representation of parts and one for the automatic retrieval of assemblies. These prototypes were intended primarily for orientation and evaluation purposes. After the evaluation of the prototypes, a functional specification of the MAGIC system was established. During the second year, the structure of the relational database used for the internal representation and the conceptual design were elaborated. The third year of the project was reserved for the establishment of the final prototype, the implementation at the industrial partners site and the development of the CAD interfaces. The collaboration of different industrial partners as well as research centres and software houses is a very good method to produce a general usable system.

2. NEW METHODS AND TECHNICAL INNOVATION

The Classical Group Technology Systems

Group Technology systems are tools that make it possible to identify and group similar parts. This implies some sort of classification. In the classical GT systems like Opitz, Miclass and Cetim-pmg the classification is realized by giving each part a codenumber. The digits of this codenumber depend on the properties of the part. Although these systems can be used successfully they have certain disadvantages :

1. Manual coding. The manual coding of the parts is rather time consuming. The different interpretations of code units by different operators can lead to ambiguities

and a possible inconsistency between the part and its code.

2. Information Degeneration. Due to the use of a codenumber a lot of information about the part is lost during coding.
 3. Rigid classification. The use of a codenumber leads to a rigid classification. This classification allows only one point of view. Therefore most classic GT systems are oriented towards one application: design or manufacturing.
- MAGIC solves these problems and provides a lot of additional advantages.

Three fases: Generation, representation and retrieval

A GT system can be divided into three main functional blocks: generation, representation and retrieval (Fig. 1).

1. Generation. The representation, to be used in the GT system has to be generated for all new parts. With the classical GT, the codification of the part is done manually or interactively on a computer system with questions and answers. Within the MAGIC approach, the generation of the representation of the part is automated.
2. Representation. A good representation is very important since it is the limiting factor for the application possibilities of the GT system. Instead of a multi-digit code, MAGIC uses a structured representation in a relational database. This representation is based on the "feature" concept and far more complete than the classical codenumber.
3. Retrieval With the classical systems retrieval is performed by setting ranges for the different digits of the code. MAGIC allows complex specifications to be formulated. One of the goals is to retrieve drawings based on a rough sketch on the CAD system.

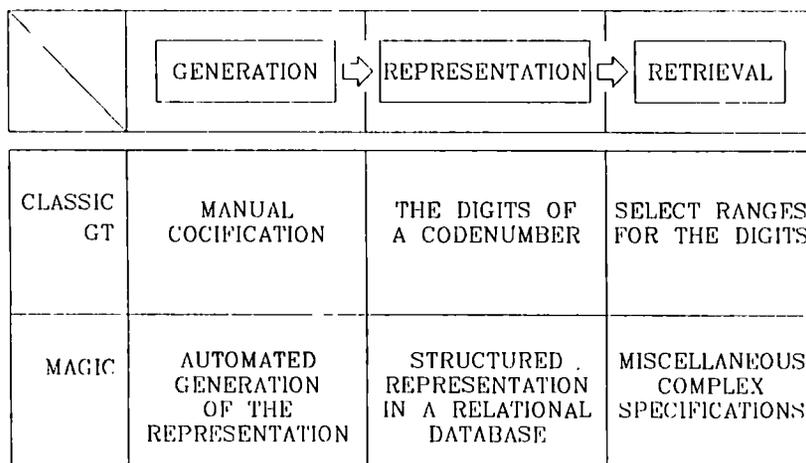


Fig. 1 MAGIC compared with a classic GT.

Comparing MAGIC with a Classical GT system

How are the classical GT problems solved in the MAGIC system?

1. Manual Coding. By automating the generation, codification work is reduced and ambiguities are excluded.
2. Information Degeneration. The information on the part that is saved is much more

complete than a classical codenumber. Moreover instead of dividing variables into classes using the digit of a codenumber, the real values are used. These real values and the presence of formfeatures are real useful to provide input to automated process planning systems.

3. Rigid Classification. The relational database structure, used for the MAGIC representation, allows to make a much better description of the part than a codenumber. The representation is independent of the application field. At all time, it is possible to add new information. The available data can be dynamically regrouped to directly answer the questions coming from different departments.

Additional Advantages

Magic not only solves the problems of the classical group technological system, but also offers a lot of new features:

Simple parts and assemblies. The classical GT systems are only suited for simple parts. MAGIC also deals with assemblies and products.

Functional Properties. In the design department, the function of a part can be an important retrieval criterion. MAGIC deals with these functional properties.

Integrated with CAD Design. MAGIC is integrated with CAD design. This integration is double. First, the information that is introduced in the CAD system is used to generate the MAGIC representation. This means that coding is no longer a separate process. It is completely integrated with the design task on the CAD system. Secondly, the CAD-user interface can be used to generate retrieval specifications. For example, the rough sketch of a part on a CAD system serves as the retrieval specification for a wanted part.

Feature Based Design. Instead of using lines, arcs and text to represent parts in CAD drawings one can use features like cylinders, holes, engineering information etc... MAGIC makes use of feature based design.

An extensive datastructure. The MAGIC part model can contain the following information:

- administrative data: the partname, the partnumber, the name of the designer, the date of the design, the scale, ... These data can be found in the drawing header. Retrieval based on these data, can already provide promising result.
- geometrical data: the overall length, width, height, diameter, the parameters of the corresponding parametric model, ...
- technological datas: material, surface quality, heat treatments, ...
- tolerances: both sizerolerances, form- and place tolerances.
- functional data: all answers to the question: what is the function of this part? Examples: the realisation of a linear movement, a circular movement, ...
- the presence of formelements: (standardised) holes, thread, grooves, pockets, cylinders, ...
- a condensed description of the general form of the part. A possible method is to work with the Fourier transformation of the contour. In the Fourier coefficients, each direction change and the size of the lines are stored in a general way. Retrieval is done by a comparison of the stored coefficients.

3. THE MAGIC ARCHITECTURE

The aim of the MAGIC architecture is to provide an open and general usable group technological system. To assure this, the MAGIC development team uses a lot of standard software tools.

The MAGIC functions

The MAGIC functions guide the user in the logical actions he has to take to input and update data. The following functions are available :

The "Master Data" function defines a unique part identification and contains a lot of administrative data like: part name, user part identification, part designer, part release date, ... Every part in the system must have an entry in this table.

The "Group-tables" contain information about the parts such as geometrical and functional data, material, ...

The "Group-structure" table defines the classification structure of the group tables. A network structure can be defined, allowing a multiple way access to the same group table. A tree structure is a simplified application.

Multiple classification is allowed. This means that a part can belong to several group tables: one specifying geometrical data, another specifying functional data.

The "Bill Of Material" function defines the relation between the physical parts: an assembly is composed out of several simple parts, a product contains a lot of assemblies.

The MAGIC Programming Interface

The aim of the Programming Interface is to provide the user with a performing and easy to use tool to develop his company dependent MAGIC applications. The PI checks the consistency of the data flow: it will be impossible to assign attributes to not existing parts, a part can not be deleted if it occurs in a Bill Of Material list, ...

The most important application of the Programming Interface is the CAD interface. This interface will enable the automatic generation of the group technological representation. All information input during the design process can be transferred automatically to the MAGIC system:

- administrative data: the name of the designer, the name of the part, the creation date of the drawing,...
- form feature data: the used form features and their relative position, ...
- technological data: surface quality, special heat treatments and coatings, tolerances, ...
- functional data: a description of the function of the part, ...

The Programming Interface can also be used to extract data from the MAGIC data base to provide input to automatised process planning systems.

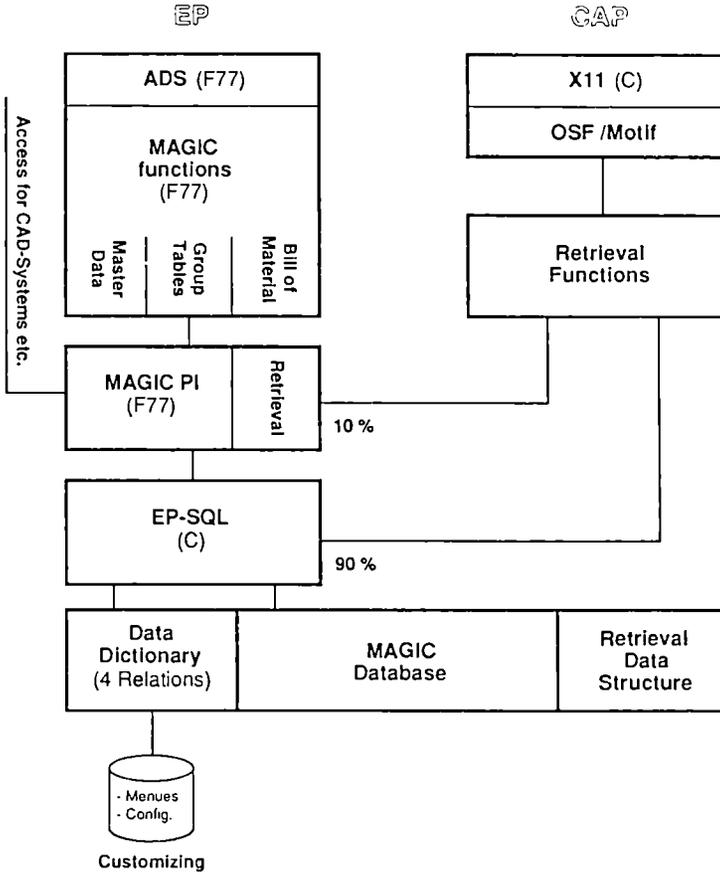


Fig. 2 The MAGIC architecture

The EP-SQL neutral data base interface

To allow the usage of several SQL based relational data base management systems (Oracle, RDB), the general SQL interface developed by one of the partners was used. This tool also assures an easy portation to the other SQL based relational data base management systems.

The MAGIC data base

A good representation is very important since it often is the limiting factor for the application possibilities of the GT system. The internal representation should be able to contain all possible useful information and should avoid information degradation. It should also be independent of the application, being suitable for design as well as for manufacturing. Therefore, instead of a rigid multi-digit code, MAGIC uses a structured representation in a relational database allowing to store the real values.

RETRIEVAL ON CLASSIFICATION DATA

Current request: Temp

Define criterias

GROUP: G234 Synonyme: ExampleGroup Description

Parameter	Length	Width	Diameter	Color	
Type	Real	Real	Integer	String	
Unit	cm	mm	mm		
Selection criterias	< 10.5				
	> 3.0				
Display in result	Yes		Yes		

Options: no display parts

Fig. 3 The Retrieval on Classification data

The retrieval functions

Retrieval is a very important aspect of the MAGIC project. Since MAGIC uses an extensive representation, the retrieval possibilities offered to the user must be powerful enough to allow the full use of this representation. It is useless to store information about form features if it is impossible to look for parts using this information.

There are two main operational levels in retrieval.

- The first application level is to retrieve previously classified parts, looking for specific dimensions. This level is based on an a priori classification: when created, the parts are assigned to a group.
- The second application level is to define new groups based on the selected characteristics. This application level is an a posteriori classification: the stored characteristics are used to define new groups. This possibility makes it easy to dynamically define new groups matching the changing needs.

A useful feature to automate retrieval is to select a user defined number of parts most closely to a given value.

It should become a habit in each company to look for existing similar parts before starting a new design. Nowadays this is often "forgotten". At the moment it takes more time to retrieve a similar existing part than to draw a new one. Magic will change this situation drastically. This will not only improve the productivity of the design department but also of the manufacturing department. If an existing part can be used again, there is no need to make a new process plan. The lotsize increases.

The MOTIF User Interface

To provide a user friendly interface and in order to develop a hardware independent software, the retrieval module uses the OSF/MOTIF user interface.

Conclusion

Through the development of an open system and the choice of several standard interface tools, the consortium wants to assure the maximal usability of the MAGIC concepts for the European industry.

4. THE USE OF FEATURES

A form feature is defined as a number of basic geometrical entities (points, lines, arcs, etc.) grouped into a logical instance associated with specific production oriented characteristics. Features represent a higher conceptual level than lines, arcs and text used in existing CAD/CAM systems because they inherently contain more information. Simple entities such as lines, arcs and circles can be grouped logically to denote a functional or manufacturable entity. The typical feature based design systems provide a set of standard engineering shapes, or features, that are ready to use. Designers simply select a generic feature, like a hole, a chamfer or fillet, and then enter the necessary values, or parameters, to generate the specific feature they have in mind.

The industrial partners see a lot of advantages in using features and assigning attributes to geometrical elements while designing on a CAD system. It is indeed a very economic and unambiguous way to build the Group Technological representation. Figure 4 illustrates this possibility.

In the example the following geometrical elements are used: cylinders, cones, key slots, gears.

In addition, some functional information is added: bearing seat, connection element, seal, shaft extension.

Using this information, it is easy to get a list of all seal diameters used in the company: One has to retrieve all cylinders with the function seal.

Another example is the use of parametric drawing programs. Although it is easy to generate new drawings with parametric programs, it still is interesting to look if such a part does not exist already. Magic provides the possibility to store those parameters in group tables. These parameters can be used to retrieve existing parts and for statistical purposes to standardise certain dimensions.

Feature - Cylinder								
partnumber	sequence number	diameter	length	dia tolerance		length tolerance		function
				upper	lower	upper	lower	
D0232016	2	55.00	66.00	+0.03	+0.01			bearing seat
D0232016	4	76.00	104.74	+0.40	+0.10	+0.00	-0.50	connection
D0232016	6	76.00	53.74	+0.40	+0.10	+0.00	-0.50	connection
D0232016	8	55.00	39.00	+0.03	+0.01	+1.00	-1.00	bearing seat
D0232016	9	55.00	35.00	+0.19	-0.19			seal
D0232016	11	50.00	110.00	+0.02	+0.00	+0.50	-0.50	shaft extension

Feature - Cone						
partnumber	sequence number	left diameter	right diameter	length	angle (degrees)	function
D0232016	1	53.00	55.00	1.00	45	shaft face
D0232016	3	65.00	76.00	24.26	15	connection
D0232016	7	76.00	65.00	24.26	15	connection
D0232016	12	50.00	48.00	1.00	45	shaft face

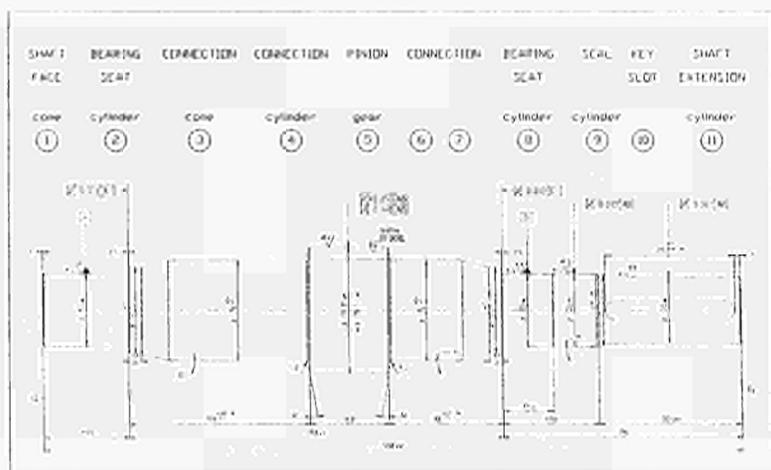


Fig. 4 The use of form features

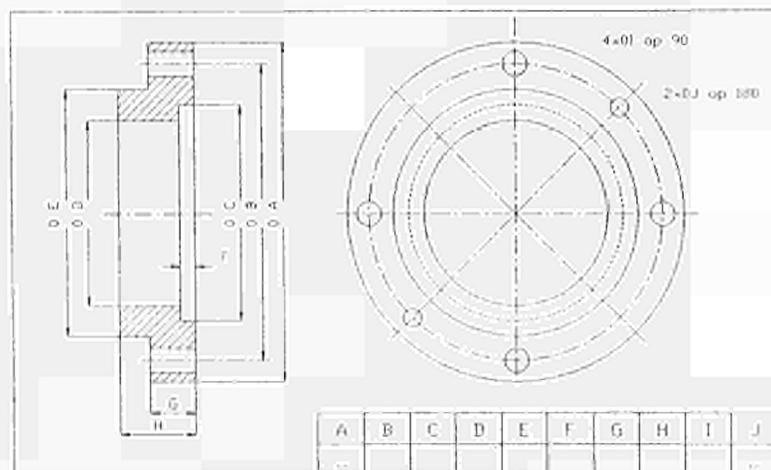


Fig. 5 Parametric drawings

In general, the following feature categories can be distinguished :

- Form features (geometrical elements) either having a functional or a machining character.
- Pattern features: specifying a repetition of features.
- Connection features: local geometrical relationships between features, mostly tolerance features.
- Property features: non-geometrical characteristics, mostly technological characteristics like heat treatments, surface treatments, chemical treatments, hardening, painting, coating, surface quality, ...
- Application features: relationships between features according to assembly, ...

To each feature, different types of information can be added: geometrical information, functional information, tooling information, cutting conditions, operation sequence information, cost information, quality control information, ...

Adding all this information to the machining features is a nice way to automate the process planning task. The easiest example is to drill a hole. Depending on the hole depth and width, the hole will be centred first, then rough bored and the finish bored. Depending on the depth, a peck-wood cycle is needed or not. Thus requiring three tools, having certain cutting conditions and a certain cost. Proceeding this way, the whole workpiece could be described. The process plan will be generated automatically. The FLEXPLAN project is researching this.

Different types of feature-feature links (compared to part-part links expressed in the Bill Of Material) can be considered:

The first understanding of a feature-feature link is several features composing a higher level feature: a tapered hole can be considered as an assembly of several hole features.

A second feature-feature link could be the connection of a form feature to a pattern feature. This expresses that the same form feature is used on different locations on the same part. The pattern feature indicates application points. It is a transformation of the feature to different locations.

A third feature-feature link is tolerance specification (connection features) between two or more geometrical features (form features): a concentricity between cylinders, a parallelism between two planes.

5. MAGIC AS AN INTEGRATING ELEMENT

The relational data base used to store the MAGIC representation, is the core of the MAGIC system. This data base can serve as a common technological data base for a number of applications in design and manufacturing, hereby acting as an integrating element. Links to computerised applications allowing input and retrieval of information can be provided.

Design

In order to speed up the design process and to add the appropriate information, CAD macro's, parametric drawing programs and standard component libraries are used. These tools can be controlled from the MAGIC system. During the use, standard dimensions are read from the MAGIC data base where as the variable dimensions are input interactively on the CAD system, depending on the actual design. Next, a new part is created in the MAGIC system and all these dimensions are assigned to it for later use.

Manufacturing

The necessary manufacturing information, available from the design department can be retrieved from the MAGIC data base. Inside MAGIC, the Bill Of Material information needed for production planning is stored. For standard parts, the name of the supplier and the partnumber is stored. For all parts to be manufactured inside the company, the dimension and the part class are available.

Although the use of Group Technology in computer aided process planning (CAPP) is not new, the MAGIC system offers new possibilities to variant CAPP because MAGIC stores not only the part class but also the real geometrical dimensions. To each part family, a standard process plan is connected. Adapting the standard process plan to the particular needs of the part results in the dedicated process plan of the part. Since

the necessary information is stored in the MAGIC database, dedicated variative process plans can be automatically generated.

The feature based representation provides the necessary input for an expert system to generate process plans (a generative system). The philosophy is used in other European research projects like ESPRIT-2457-FLEXPLAN (Knowledge Based Planning and Control in Manufacturing Environments) and BRITE-3480-IDEFIX (Ideal tools and methods for fixturing).

A lot of other applications can also take benefit from the MAGIC system. In scheduling similar parts can be grouped and produced together avoiding set-up and tool changes. Parametric NC programs can be elaborated, reducing the programmers effort to the introduction of a reduced number of parameters.

6. THE MAGIC PROTOTYPE

At the moment, the alpha test release of the MAGIC prototype is installed at one industrial partner site. The system contains already about 7.500 parts, automatically fed from the CAD system. When starting a new design, the designers systematically consult this data base to look for existing similar drawings. This attitude avoided the redesign of a number of similar parts, providing savings in all subsequent departments. Now, the company is working on the link to their process planning software. They are very satisfied with this prototype and they expect a lot of benefits from it.

During the last months, the industrial partners have been preparing their CAD environment to install the prototype. They are looking forward to install the beta release of the MAGIC prototype which comes available in August 91. Real life test results are expected from September on.

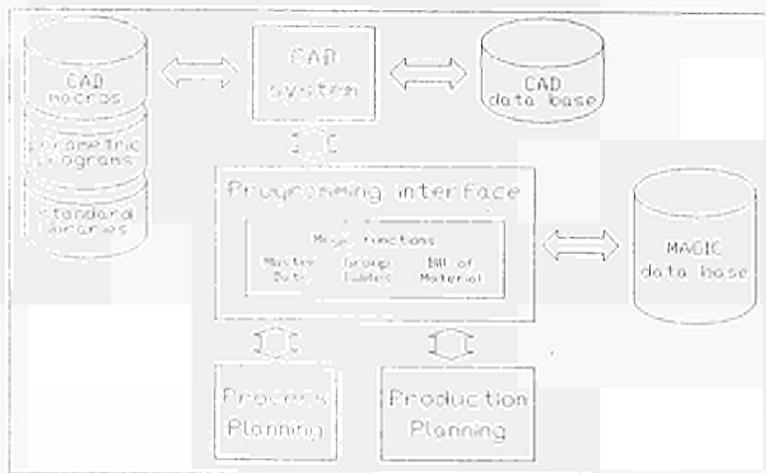


Fig. 6 MAGIC as an integrating element

7. EXPLOITATION OF RESULTS

The final result of the Magic project, actually in full development and allowing the above described features, will be a Computer Automated Group Technology software.

The information technology partners will commercialise directly both the consultancy and the software results of the project. Eigner & Partner has already 300 customers

(1200 installations) in the field of GT and CAD/CAM and intends to further develop the MAGIC prototype to commercialise it as a full product. CAP SESA Industry will use the Magic core as a basis for customer specific projects.

The industrial partners N.V. MICHEL VAN DE WIELE, LVD and C.M. MARES will make direct use of the results by applying them in their own manufacturing.

The research centres will exploit the results by giving seminars, individual sensibility, searching new application fields and in general promoting the use of GT. CETIM has about 12.000 French and WTCM/CRIF about 1.800 Belgian industrial members.

8. BIBLIOGRAPHY

Richard E. Billo, Rob Rucker, Dan L. Shunk, "Enhancing Group Technology Modeling with Database Abstractions", Arizona State University, Tempe, Arizona.

Richard E. Billo, Rob Rucker, Dan L. Shunk, "Integration of a Group Technology Classification and Coding System with an Engineering Database", Arizona State University, Tempe, Arizona.

Esfandiar Kamvar and Michel A. Melkanoff, "Automatic Generation of GT Codes for Rotational Parts from Cad Drawings", UCLA, California USA

Bardwell Salmon, "Group Technology: The Key to Integrated Manufacturing", OIR, Waltham, Massachusetts.

U. Beckendorff, C. P. Brück, "Knowledge based planning and control in manufacturing environments" Reports on results and progress of ESPRIT project 2457, 1989

H.K. Tönshoff, U. Beckendorff, N. Anders, J. Detand, "A process description concept for process planning, scheduling and job shop control" Preprints of the 22nd CIRP Seminar on Manufacturing System, 11th - 12th June, 1990, University of Twente, Enschede, Netherlands

Peter Schoonjans, "MAGIC: A new approach to Group Technology" Proceedings of the 6th Esprit CIM-Europe Annual Conference 15-17 May 1990 Lisbon, Portugal p 261-272 ISBN 3-540-19616-1

INFORMATION EXCHANGE SYSTEM

THE COSINE PROJECT

Mr. D. R. H. Davies,
Director, COSINE Project Management Unit
RARE Secretariat
PO Box 41882
NL-1009 DB Amsterdam
the Netherlands

Dr. D. Goodman
Department of Computer Science
University College London, UK

Mr. G. Knight
Level-7 Ltd
Bracknell, UK

Mr. J. Romaguera
BUESS Computer Network Consulting
Berne, Switzerland

SUMMARY

COSINE (Cooperation for Open Systems Interconnection Networking in Europe), is a Eureka Project which has been established to provide the European academic, industrial and governmental research community with a computer networking communications infrastructure based on OSI protocols. The implementation of the Project is being carried out by RARE (Réseaux Associés pour la Recherche Européenne) which has established the COSINE Project Management Unit to carry out the work on its behalf. The Project includes a number of sub-projects and the provision of pilot services to the user community. A number of these sub-projects and services are now in place and have a direct impact on the working environment of the research community. They bring European-wide connectivity in electronic mail, directories and information services, expanding on the services already available at local and national levels. Further sub-projects and services are in the process of being established. The aim is that the services will be self-sustaining by the completion of the COSINE implementation phase.

1. Project Overview

The COSINE Project began with a Specification Phase which was undertaken by RARE (the association of European Research Networks and their users) and which concluded in the autumn of 1988. During 1989, work began on the Implementation Phase of the Project. This is also being undertaken by RARE which has established the COSINE Project Management Unit (CPMU) to carry out the work on its behalf. The implementation phase of the COSINE Project includes a number of sub-projects and the provision of a number of pilot services to the user community in the COSINE member states: Austria, Belgium, Denmark, Finland, France, Germany, Greece, Ireland, Italy, Luxembourg, the Netherlands, Norway, Portugal, Spain, Sweden, Switzerland, the

United Kingdom, and Yugoslavia. The Project as a whole is under the control of the COSINE Policy Group (CPG) which represents the interests of the participating countries. The Commission of the European Communities (CEC) is also a major contributor to the Project and is represented on the CPG.

The COSINE Project has three central aims. The first, of particular interest to the Esprit community, is to develop a pan-European infrastructure to provide a platform for national research workers to communicate with one another using Open Systems. The second is to provide the basis for specific projects aimed at facilitating the introduction of Open Systems Interconnection (OSI). The third is to ensure that the infrastructure that is established becomes self-sustaining by the end of the COSINE Implementation Phase. Details of individual COSINE activities are given below.

1.1 FTAM North American Gateway

The Pilot Gateway service will be implemented according to the following parameters:

- an FTAM system built on the ISO Development Environment (ISODE) developed by Performance Systems International Inc and others;
- a DEC system 5000 model 200 with the ULTRIX operating system;
- DECnet Router to X.25, and TCP/IP to X.25 gateways;
- management functions running and operated on Personal Computer Workstations;
- a transatlantic link to be established in the first instance by negotiating the best possible arrangement with the current academic operators of transatlantic links;
- provision of a Help Desk to assist users.

The contract, which has been awarded to Systems Ideas SA/NV located in Brussels, provides for the establishment and operation of the Pilot Gateway service over a period of one year until the end of December 1991, with an option to continue the service operation for a further year. The project will consist of a service establishment phase followed by a pilot service operational phase which initially will be limited to a set of Beta Users and gradually extended to the full COSINE community.

A service promotion activity will be carried out during the first year and will take the form of an initial service announcement notice distributed to the COSINE community via national organizations. This will be followed every month afterwards by a service notice together with a status report. One of the objectives during this first phase of the project is to determine whether the option to extend the service for another year should be taken up and, if so, what the level of service should be.

1.2 Directory Services

During the COSINE Specification Phase a study was performed on the future of directory services in the COSINE community, based on the current directory standards (CCITT X.500/ISO 9594) and a global analysis of their applicability to the requirements of COSINE. This study described the requirements and functionality of a future COSINE directory service, an overview of existing directory services, relevant concepts and status of the standards, and key problems relating to the realization of a directory service within COSINE.

The directory services sub-project will investigate and set up a pan-European pilot X.500 directory service, in collaboration with national directory pilot projects, and as a result deal with interoperability issues.

The contract for the sub-project was awarded to a consortium of funded and unfunded partners led by University College, London (UK). The other funded partners are ULCC (UK) who will operate the pilot service, X-tel Ltd (UK) who will carry out software development, and PTT Telecom (the Netherlands) responsible for PTT liaison. Representatives of a number of European national research networks are unfunded partners who have expressed a willingness to contribute to the project by taking and evaluating the pilot service. Details of the project are given in Chapter 2, "PARADISE: the COSINE X.500 Pilot Service".

1.3 Support and Information Services

This sub-project will set up a pan-European information service, facilitating cooperation between national information services and offering information (for example on products, national networks and contacts, and on international projects) to the European networking community. It also requires the provision of support tools to assist information providers in supplying and maintaining information as well as to assist in the creation of new information servers.

The contract for the sub-project was awarded to Level 7 Ltd (UK) who are providing OSI and project management expertise, as well as running the service. They have sub-contracted part of the work to Université Libre de Bruxelles (Belgium) who are providing expertise in the development, integration and testing of OSI software, as well as maintenance of the system. Details of the project are given in Chapter 3, "The COSINE CONCISE Information Service Project".

1.4 Support of International User Groups

During the COSINE Specification Phase, it was recognized that simply establishing an OSI-based networking infrastructure and associated services for all European researchers, one of the principal objectives of COSINE, would not be sufficient. It would also be necessary to encourage users to use the network and its services.

This sub-project will be undertaken to provide an effective way to promote the use of COSINE services by a representative set of international user groups that will benefit from migration to electronic communication based on OSI.

The general objectives of the sub-project are:

- to provide a general framework for the support of international user groups by developing the approach on selected, target Special Interest Groups (SIGs).
- to promote and publicize COSINE services to the European research community.
- to encourage the use of COSINE services for enhanced group communication and so improve the quality of European collaborative research, enabling interaction between disciplines without locking users into any one physical network.

The execution is planned in two phases. The first phase will be a consultative one in which a review of existing user groups already using electronic mail and other network services will be undertaken with a view to learning from their experiences. This will result in recommendations for a support structure for international user groups. In addition, a survey report will be produced describing which existing international SIGs could best benefit from the sub-project's activities, so being identified as potential target SIGs for the second phase. The second phase will then be the implementation phase, the result of which will be the establishment of an international user group support structure that will enable the project to communicate with and gain feedback from the target SIGs selected as a result of the consultative phase reports. It is intended that the execution

of the sub-project's activities can be focused on satisfying the requirements of the SIGs on the one hand and the objectives of COSINE on the other. The first phase of the sub-project started in June 1991 under a contract awarded to Logica BV (the Netherlands).

1.5 Migration of Users to OSI Services

Discussions have revealed a growing need within the user community for connectionless networks services in addition to the existing connection oriented services. The simplistic specification of the requirement for a 'multi-protocol' service is that, at any access point, users should be able to choose the most appropriate of three network services: X.25 connection oriented; TCP/IP (the de facto connectionless standard in common use at present, particularly over LANs); and ISO-CLNP, the new set of OSI standards for connectionless operation which, amongst other things, is intended to resolve some of the routing and addressing problems which arise when IP is used over wide areas. At this stage of the development of ISO-CLNP, it is impossible to specify a service to the level of detail required in an invitation to tender to commercial suppliers. There are also many questions relating to interworking, for example between applications which use different network services at different locations, which will need to be resolved before any multi-protocol service can be brought into operation.

There are however, a number of organizations making plans for an ISO-CLNP pilot project which would be used to test newly developed commercial products and to gain experience with this type of network. In cooperation with these organizations, the CPMU has organized a sub-project aimed at exploring the migration issues which arise from the use of ISO-CLNP. The sub-project will set up a pilot network that consists of a collection of ISO-CLNP capable end systems (ES) connected to both existing LANs and directly to existing X.25 WANs. The LANs will be interconnected via WANs (both X.25 and point-to-point) with ISO-CLNP intermediate systems (IS). OSI applications such as FTAM (file transfer) and MHS (message handling), as well as proprietary DECnet facilities such as SET HOST and COPY, will be tested. The proprietary applications are important in the early phases of the pilot as the number of pure OSI end systems will be too small to generate enough real data on the network.

From experience gained from building and running the pilot network, the project will:

- show the possible interoperability and configuration problems related to different routing technologies (ES-IS, IS-IS, as well as both static and dynamic inter-domain routing);
- report on the performance of ISO-CLNP over different network technologies (LANs, X.25 and point-to-point networks);
- deliver a recommendation on how the OSI Routing Framework (ISO TR 9575 administrative domains, routing domains and areas) and the corresponding NSAP address structure is best mapped into the European research network architecture;
- propose how to organize and coordinate the operation and management of large scale ISO-CLNP networking in Europe in order to avoid several independent efforts by different user groups.

The pilot network will also provide a test-bed for interworking experiments involving transport service bridges and network tunnels although such experiments themselves are not part of the sub-project.

1.6 FTAM Product Interworking

The conclusion of the COSINE Specification Phase was that, although most of the conformance testing services required by COSINE would be available at the time of the COSINE Implementation Phase, few interoperability testing services would be available or planned. This sub-project will therefore subject conformance tested FTAM products from different vendors to interoperability tests for a trial period with the intention of exposing any remaining incompatibilities. If necessary, the sub-project will also arrange for the products to be conformance tested by existing European test centres beforehand. The sub-project has been the subject of a recent invitation to tender and contract discussions are under way.

1.7 European Full Screen Pilot Project

ISO started standardization activities in this area in the early 1980's but unfortunately underestimated the complexity of the standardization task. Additionally, there has been confusion surrounding the need for ISO-VT (virtual terminal) in light of the anticipated international standardization of the X Window System (X Windows). In fact, ISO-VT and X Windows are complementary rather than competitive as they each possess properties which are advantageous in different circumstances.

An objective of this sub-project is to demonstrate the integration and complementary nature of the two techniques. The sub-project will draw upon the now stable ISO-VT documents that have been developed by ISO and the European Workshop on Open Systems (EWOS) and will also pay particular attention to windowing environments which are emerging in the ISO standards.

The sub-project will adopt a two phase approach. The first phase is to be fully funded by COSINE. During this phase, a portable pilot implementation of an ISO-VT kernel will be produced that can be transferred to a range of hardware platforms that are commonly used in European research establishments. The second phase of the sub-project will encompass the porting of the kernel to specific hardware platforms. This will require a limited amount of COSINE funding that will be augmented through manufacturer involvement and contributions. In this second phase, interworking tests will be carried out. Upon successful completion of the tests, products produced by the COSINE funded part of the sub-project will be made freely available to members of the research community.

It is intended that in the second phase of the sub-project, an implementation of a host ISO-VT user will be realized in an information system forming part of the COSINE Support and Information Services sub-project, and that continued funding for maintenance of the kernel and documentation for a period after the end of the project will be found from the commercial sector. The sub-project has been the subject of a recent invitation to tender and contract discussions are under way.

1.8 X.25 (1984) Service Provision

A basic element of the COSINE Project is the international X.25 backbone data communications network (IXI). This was implemented to provide X.25 connectivity to the European research networks which previously had to rely on the low performance inter-connectivity of public X.25 networks. The IXI project offers sub-network connections for the transfer of messages between national systems. At the time of writing (May 1991) work is in hand to uprate the capabilities of the IXI network and procurement of

a new network infrastructure which allows for the possibility of offering 2Mbit/s access and X.25 (1988) is nearing completion.

1.9 Message Handling Systems

An electronic mail service available to the research community throughout Europe is considered to be essential. National X.400 services are already available to research workers in many of the COSINE member countries. The purpose of the COSINE Message Handling System (MHS) service is to ensure interworking between these national services (and to assist with the creation of, and migration to, national X.400 services in countries where they do not exist). The COSINE MHS service is built on the functionality which had been provided by the RARE MHS pilot project, established in 1988. From the initial provision of a minimum support activity required to satisfy basic end-user needs, an upgraded international integration and support activity will be provided to create a single European OSI MHS service, achieved by coordinating national activities.

The managers of the national X.400 services are seen as the primary channel through which progress towards satisfactory international interworking can be achieved by COSINE MHS and they are being advised on the resources required to operate an international service.

As there are currently no suitable public services for gatewaying between the X.400 based MHS services of the COSINE community and the non-X.400 based electronic mail of the research community in the USA, it has been recognized as essential to provide such a gateway service during the COSINE Implementation Phase to support communication between the research communities. A number of MHS gateways already exist at the national level. This COSINE service will establish a backup service for these gateways, as well as provide a service to users on those national networks that do not yet have a gateway. Additionally, the gateway service could be developed and extended to research communities other than those identified in the plan. An initial investigative phase has established a service definition and identified suitable software products.

COSINE MHS is complementary to the Y-Net initiative being undertaken by the CEC in conjunction with European manufacturers.

The contract for the interworking of existing X.400 domains was awarded to SWITCH (Switzerland) with their subcontractor BUSS Computer Network Consulting (Switzerland). A twelve month contract to establish the gateway service was awarded to AREA per la Ricerca (Italy) with their sub-contractor INFN (Italy). More details are given in Chapter 4, "The COSINE X.400 MHS Service".

1.10 Major Achievements

The major achievements of the COSINE Project so far have been:

- management and coordination of the work of a team distributed widely in Europe;
- development and operation of procedures for tendering and for tender evaluation;
- successful negotiation of sub-contracts and start up of five COSINE sub-projects and services;
- preparation of plans for four further sub-projects;
- specification and successful promotion of a COSINE contribution to a new ISO CLNP pilot sub-project;
- introduction and successful exploitation of the IXI pilot service;

- preparation and issue of an invitation to tender for the IXI production service.

1.11 Next Steps

To date the work of the COSINE Project has concentrated on the specification and implementation of a set of activities aimed at demonstrating the viability of OSI and at facilitating its use in the European research community. The next steps, now that the initial projects are delivering results, is to encourage the wider exploitation of them within the broader research environment. Current users tend to be computer literate. A bigger challenge will be to make the benefits of OSI available to the less experienced user. User awareness programmes and the development of support activities to encourage new users are important aspects of the ongoing work programme. Another area of work is that of management. The research community is a good example of a multi-domain environment. Experience with the current projects has shown some of the operational and technical management challenges. It is proposed to establish a project to examine how the emerging OSI standards in the area of management could assist in simplifying the need for human procedures to perform technical management functions.

2. PARADISE: The COSINE X.500 Pilot Service

2.1 Introduction

A distributed electronic directory is a network's window on itself and its user community. A network of directories reveals who and where its users are and how their organizations are structured. It can also show its users how to get in touch with each other and even what they look like. The PARADISE (Piloting A ReseArchers' Directory Service for Europe) project is COSINE's pilot international directory service and is an essential component of the OSI infrastructure for the European research and development community of academic, commercial and governmental organizations. Directory services are seen as essential for the spread of electronic mail and a necessary tool to make Europe interwork electronically.

The project has two major components. The first is the coordination of existing, experimental, national pilot directory activities; the second is the development of directory servers that can be replicated and used where no national pilot activity is taking place. The experience gained will enable the CPMU to issue a tender for a self-sustaining operational service in 1993.

2.2 Timetable

Earlier this year PARADISE announced an operational service providing a central configuration DSA (Directory Service Agent) with connectivity via both X.25 over PSPDN and the Internet using RFC-1006 over TCP/IP. This DSA contains the "root of the world" node and provides the glue at the top of the international directory tree (DIT). During the summer a central DUA (Directory User Agent) with public access was made available so that any user with a remote terminal/PC terminal emulator can look up information in the directory. Multilingual versions of this interface will also be made available through the project. These central services are provided by ULCC who offer support and a help desk to appropriate users. Coinciding with the latest full release of the ISO Development Environment (ISODE) software, an enhanced DSA together with the DUA developed for the pilot, was packaged and made publicly available, together

with management tools provided to manage user accounts and DIT entries. At the end of the project there will be an evaluation of the metrics of the service both nationally and for full directory implementation as well as developing recommendations on how future X.500 standards should be improved. By the summer of 1992 those offering OSI services should be able to use the pilot directory service as the basis for information collation.

2.3 Interoperability

Until very recently national directory pilots both in the United States and Europe have been using ISODE QUIPU software which was developed at UCL through the Esprit I INCA Project and subsequently enhanced by UCL and X-Tel with funding from the UK JNT (Joint Network Team). In the early phases of establishing standards and developing pilots this was highly expedient but with the growing requirement for X.500 directory services it is only to be expected that other implementations will be developed. The PARADISE project is committed to incorporating new and different implementations into the pilot and to demonstrating the interoperability of different protocols. Earlier this year, the first non-QUIPU implementation in the pilot was run on an experimental DSA in Italy. The national pilot in France, OPAX (Operation Pilote X.500 dans le cadre Aristote), is using the PIZARRO implementation developed at INRIA which evolved out of the Esprit THORN project. By the end of the pilot phase it will be surprising if there are not at least another half-dozen X.500-based implementations available and in use both in Europe and North America.

2.4 International Co-ordination

Of the eighteen COSINE countries involved, most are now registered in the DIT and, for those that are not, the project offers to run DSA facilities until such time as a national pilot is started. The PARADISE project is actively coordinating the development of national directory pilots by creating a forum for the discussion and transfer of experience and support. Collaboration will take place on confronting the existing pilots with the problems of scale both in maintaining functional standards such as extending the RARE/IETF naming architecture schema as well as encouraging growth of the directory. In parallel with the project's liaison with the PTTs, an important role of the national pilots is to manage and monitor levels of usage in order that a realistic assessment of the parameters of running an operational service can be made. In addition, close contact is being kept with piloting activity in North America and the rest of the world as a result of which PARADISE has recently produced the second of four international reports on X.500 directories. It is to be hoped that before the end of the project it will have been possible to have included reports from East Europe, South America and the Far East, making the directory truly global.

2.5 Local Management

The model for the COSINE pilot is of a distributed directory. The advantages of this approach are that local data can be managed locally, a requirement important to most organizations for reasons of security and data reliability. Whilst encouraging the growth of local networking infrastructures, insufficient technical expertise with distributed directories can create a strain in distributed monitoring and maintenance. The use of local DUAs can provide a much higher level of performance and quality than is possible

remotely, particularly as the bulk of directory enquiries tend to be local. Hardware and software vendors can be used to handle local support issues and not affect the scaling to a full European Directory service.

2.6 National Administration

The COSINE pilot is geared towards the research community: to date most of the interest in X.500 interconnection has come from the academic community with most national pilots being run by dedicated workers in universities and research institutes across Europe. In a fully realized directory, the PTTs are expected to play a significant role in coordinating national activities and ensuring reliable levels of service and connectivity. The major issue for the PTTs is whether X.500 can be used as the basis of an operational commercial public service. Until now there has been no requirement for establishing accounting mechanisms which are essential once centralized funding of piloting activities cease and both the academic and non-academic communities become committed to using the directory. Of considerable importance in the assessment of the overall viability of X.500 is the investigation of the exact relationship between X.500 and the TPH028 developments for interconnecting public telephony directory services which have been extensively tested amongst the PTTs.

2.7 User Groups

It is still difficult to tell how much real, as opposed to experimental, use is being made of the directory. There appears to be a gradual increase in usage as public access DUAs become more widely available and better known. Suitable interfaces are seen as critical in most countries: the UK, Spain, Norway and Sweden have developed a variety of interfaces for use on different platforms. The PARADISE project is providing a central DUA based at ULCC with public access. This DUA, which will also be available as an off-the-shelf package, is intended for the inexperienced and naive user. In particular, the central DUA service is intended to cater for organizations, such as SMEs (small-to-medium size enterprises), with no resources for a local directory or access to a national facility. The project has already had outline discussions with the Y-Net project which is expected to start its own X.500 programme in April 1992.

The sub-project is in the process of conducting a series of surveys in a variety of sites across Europe, targeting specific user groups. This is to establish some critical feedback on interfaces, response times, availability and the viability of the directory as a whole. The national pilots participate in probing DSA usage and availability which are then mapped onto networking logs to give an overall picture of DSA availability. The five different classes of attributes specified by Steve Kille [1] to give an indication of the level of maintenance of DSAs are being deployed to give both managers and users an indication of the quality of service, that can be expected from any DSA.

In many cases different organizations may "share" the same DSA at either local or national level and pay an appropriate administrative charge. This facility is also available at the European level but it is in the interest of an organization itself as well as the functionality of the directory as a whole for a node to be sought at an appropriate national or regional point in the DIT.

Crucial to the success of PARADISE is the demonstration that user requirements have been satisfied. This depends upon establishing a critical mass of entries leading to a demonstrable increase in real levels of usage. This can only be achieved once the sub-project can promote stable and efficient access with relatively simple implementa-

tion procedures and user-friendly interfaces. The success of the other COSINE sub-projects and services is of enormous value in stimulating confidence in user communities to make use of the COSINE infrastructure.

2.8 Applications

The European directory is committed to providing "white pages" access only but with facilities to incorporate the directory transparently into FTAM and X.400 usage. Following on the experimental work taking place in North America, "yellow pages" lookup should be possible within two years. The use of directories as a promotional tool was established in pre-electronic days but through the use of more sophisticated interfaces this facility has already been considerably enhanced by companies providing logos at the organizational level and personal entries including photographs and even voice messages.

3. The COSINE CONCISE Information Service Project

3.1 The CONCISE Project

CONCISE (COsine Network's Central Information Service for Europe) aims to provide a pilot pan-European information service to the COSINE community, based on an open system environment and accessible by OSI protocols.

Many countries already have national information services. The aim is to provide a service that will complement these national services by providing information Europe-wide and by providing information about other existing services. Thus it will form the central focal point for users to obtain information about networking, projects, products, and services, as well as about COSINE itself.

The sub-project also aims to distribute the software produced, so that the service can be duplicated by existing and by new information services. International special interest groups will be able to use the tools provided by the project, either by setting up their own server or, more likely, using the facilities on the central server or one of the national servers. Close liaison will be maintained with other COSINE activities, especially so that international special interest groups can specify the tools which should be provided for their use by the CONCISE project.

The knowledge gained from running the pilot service will be used to study the feasibility of providing a self-supporting information service, as well as recommending ways in which the service might evolve.

3.2 The Central Information Service

The philosophy of the CONCISE server is that it will be the central focal point for all information. Once researchers have access to the information on the CONCISE server, they will be able to find information about what is available to them. In other words, the CONCISE server is the starting point that will lead them on to other information which is perhaps more directly relevant to their own work; it holds information about information: meta-information.

The CONCISE server will be located at the offices of Level-7 Ltd. in Bracknell, UK. It will have a 64Kbit/s link to IXI through the UK academic research network, JANET, thus providing access for all users in the COSINE countries. As a pilot, the server will be capable of supporting at least twenty simultaneous interactive users, as well as X.400 electronic mail and FTAM file transfer access. The service will be supported by help

desk staff who will assist users who have problems in accessing or using the service. They will also take note of any comments made about the service in order to improve it. Comments and questions will be accepted by electronic mail, telephone, letter, facsimile or as entered during an interactive session by a user.

The individual items of source data will be stored in a set of files. These files will be accessible using FTAM. Provided the filenames are known, an FTAM responder will allow the users to take copies onto their own machines. A relational database will be used to store other information such as:

- a title
- source of the information
- date of original
- date of the last change
- keywords for searching
- a brief summary
- the type of data (text, binary, ODA, programs)
- the size of the data.

As well as these, there will be structural information, indicating the hierarchical structure as seen by the users. The summary information will give the user a chance to see what the information is about without looking at the file itself. For interactive users, for example, it may not be possible to display a bitmap file, but they need an indication of what the file is about before downloading it to their own machine.

The first service to be introduced will be X.400 electronic mail. Queries to the database will be set out in the body of an X.400 message. The server will then interpret the instructions, obey them and return the results. Each query, of which there may be several in one message, will have a result. Each result will be placed in a separate body part of an X.400 message and then returned.

Later, interactive access to the server will be available either through virtual terminal (ISO-VT), X.29 or dial-up. A user interface will be developed which will allow the users to browse the database with a series of menus based on the hierarchical structure that will allow the users to make their way through the structure one level at a time if desired. The menus will be constructed dynamically from the titles of the nodes in the database. For the more experienced users, the full set of commands will allow immediate access to any item of information, without the need to use menus. The set of commands available to the users will be almost identical to the set of commands available to the X.400 users, thus providing consistency for those users who use both methods.

Figure 1 shows the architecture of the system in terms of major modules.

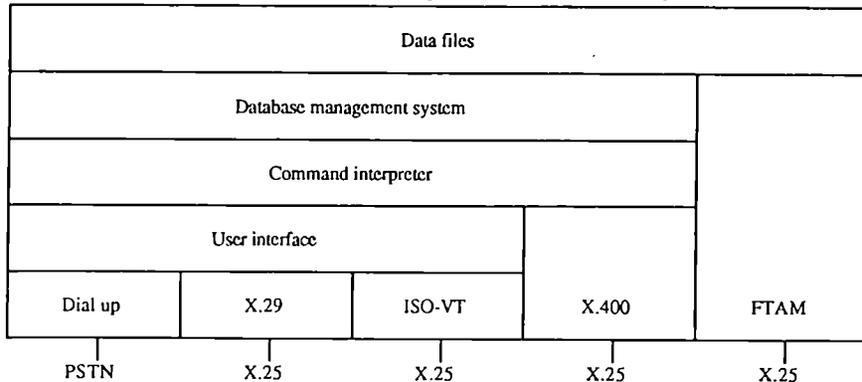


Figure 1: Module Architecture of the Server

The data files will be accessible directly by an FTAM responder that will be running at all times. For the other forms of access, a command interpreter is available which interprets and obeys commands given by the users and then returns the results. X.400 access will be a user agent to which queries are addressed. The set of commands in the body of the message will be passed to the command interpreter. The result set will be passed to the X.400 access module for the reply.

For interactive access the user interface will accept commands from the user and pass them on to the command interpreter, as well as displaying the results of the queries on the screen. Using menus, or by giving a direct reference to a particular node in the hierarchy, the users will be able to browse through the information and then read it. If the route to a particular node is not known, or is not found by browsing, then a keyword search may be performed using boolean logic. If the information is IA5 text then the interactive users will be able to display it directly on the screen, otherwise they will be able to obtain a file name and get the file by either FTAM or X.400.

General users, who have read-only access, will not have to login to the system. Information providers, who will have a portion of the information hierarchy to manage, will be able to add, update or delete information. They will have to be registered by the system manager and login before they can use these services.

If a piece of information is updated regularly, users may wish to obtain the updated information automatically. They will be able to register their X.400 address with a particular node in order to get any updates automatically. Similarly, they may wish to get any new items in a particular place in the hierarchy, which they will also be able to register for. Some information providers with regular updates may wish to set up an automatic process to do this. These users will be able to leave an updated file in a known place and have it collected on a regular basis. They will have to register all the details with the information server and provide an FTAM responder to transfer the file.

3.3 Plans

Currently only access by X.400 electronic mail is available, but interactive and FTAM services are expected to be launched during the first quarter 1992.

Representatives from the national networks will be consulted as to their requirements for the service. A series of meetings are planned throughout the project at which the current status and plans will be discussed. During the later part of the sub-project, a study will be undertaken into how to develop the system into a self-sustaining service and how the system should evolve to enable this.

The software written during the sub-project will be made available to the national information providers, research institutes and special interest groups in order to allow them to set up similar services on a national or group basis. The system will contain some commercially available software, notably the database, and the communications software for FTAM, X.400 and ISO-VT. These will have to be licensed by the sites where copies of the system are set up.

As the service is to act as the focal point of information for the COSINE community, the CONCISE publicity campaign run throughout the course of the project is crucial to its success. Targets for publicity material include the press, conferences, exhibitions, as well as articles in journals.

4. The COSINE X.400 MHS Service

4.1 Introduction

COSINE MHS is an electronic mail service aiming to meet European researchers' messaging needs. It has been operational since 14 January 1991, when the hand-over from the previous RARE MHS pilot project was successfully completed. It builds on the work of the pilot, providing increasing international integration by managing the interworking of existing X.400 services. Additionally, a gateway service to non-X.400 systems is being made available.

4.2 COSINE MHS's Users and Their Needs

The end users of the service comprise any research worker, in any organization, be it academic, industrial or governmental. Successful European research is furthered by the exchange of information between researchers working in the same field. Therefore tools and services that ease this information exchange and subsequent use of the research results are highly desirable. An efficient electronic mail service is a proven way to meet the researchers' need for the speedy and reliable exchange of information.

4.3 COSINE MHS Service Structure

COSINE MHS's view of the messaging world is based upon co-operation and coordination between the COSINE MHS Project Team, which is responsible for the international coordination of the COSINE MHS service, and the various different local and national MHS organizations, through a structure shown in Figure 2.

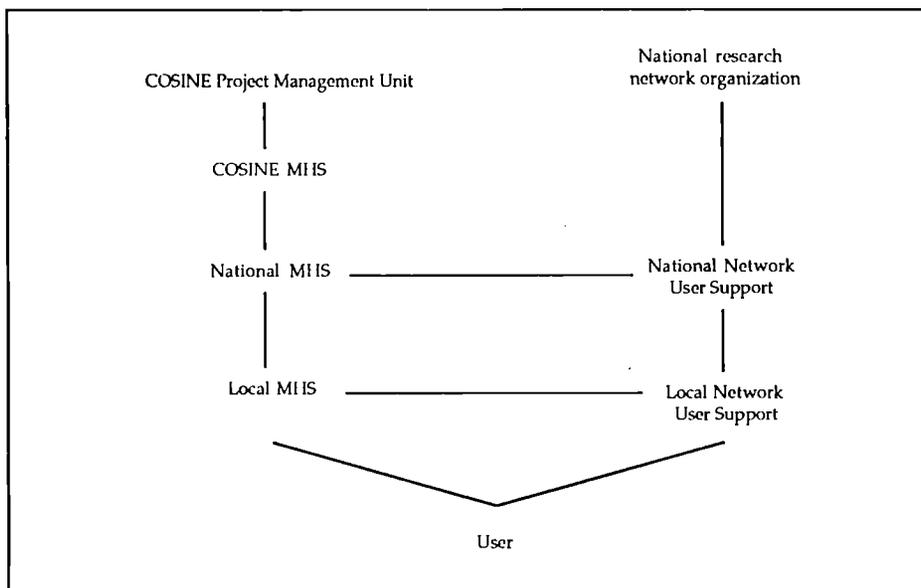


Figure 2: European MHS Organizational Structure

The local MHS organization is normally exposed to, and directly supports, the end users. Groups of local MHS organizations are represented at the international level by

a national MHS organization. The COSINE MHS Project Team acts as a central coordination point between the various national MHS organizations. Taking Switzerland as an example:

- the Swiss national MHS organization is SWITCH.
- the ASEA Brown Boveri research department in Baden operates one of the local MHS organizations coordinated at the national level by SWITCH.

A tangible result of such coordination efforts is the drive for a higher quality of service for end-users. In particular, a number of operational areas have been identified:

- training;
- user support;
- fault management;
- configuration management;
- performance management;
- accounting management;
- security management.

Procedures and agreements in all these areas are being put into place. This provides a basis for delineating lines of responsibility when issues such as fault management cross organizational boundaries, as they invariably do in such a cooperative venture. These procedures and agreements are already proving to work successfully.

4.4 COSINE MHS Services

As the project team coordinates what is in effect the international backbone service, the COSINE MHS services are directed primarily at the national MHS service operators although direct user contact is also a component of the international service provision.

Training

Customized courses and seminars on common operational problems focus on the national MHS service operators. Specialized tutorial papers are also being produced. The national operators are able to re-use the material in local training programmes. This arrangement has distinct cost benefit advantages such as:

- optimization of scarce people resources;
- local language support;
- re-use of course and tutorial material;
- building up the cooperative mechanisms essential to the running of the service.

End User Support

The project team seeks to increase the awareness of the service and the benefits of such a service to the users via articles, attendance at conferences (such as Esprit) and user group meetings. To provide a follow up to user enquiries a database of national MHS support centre contact points is maintained which is retrievable electronically. End users initially contact and seek assistance from their national MHS support centres and COSINE MHS acts as the fall-back point of contact for service inquiries in case of problems.

Fault Management

Pro-active monitoring of the network allows for the early detection of faults. An error reporting service ensures the prompt and efficient resolution of faults by the appropriate

national or local MHS organization, whilst a help desk ensures a common contact point for the registration of any faults.

Configuration Management

Vital information for the everyday running of the service (for example: Message Transfer Agent (MTA) machine configuration details, products in use in the service, etc.) is stored in a central electronic database. Configuration information objects needed for the operation of the service continue to be specified as and when they are needed. These definitions serve as a common interchange format for the dissemination of the configuration information throughout the network and allow automatic equipment updates to be performed. The project team regularly collects, reviews and updates data whilst ensuring the consistency, accuracy and suitability of the information.

Performance Management

A priority is the generation of tools that allows the measurement of parameters defined in a quality of service (QOS) agreement. These tools take statistical information from MTA machines as an input and produce statistical outputs of QOS parameters. They allow the project team to analyze problems or projected bottlenecks which could cause a degradation of performance and to take corrective action.

Accounting and Security Management

As the service is mandated to become self-sustaining, a model of how the service will account for its use is being defined. Security concerns within the service have not been a high priority to date but they are expected to assume a higher profile as the service is increasingly used for confidential matters.

Activity coordinated	Minimum achievement	User's benefit
service agreements	The responsibilities between the different MHS organizations are known	No "passing the buck"
service levels	E-mail gets delivered within a certain time	You know when to complain
help desks	A contact point that will follow up operational matters and errors reported to it	You know who to complain too
pocket handbook	A way to get started with e-mail and an easy reference guide after starting	less need for e-mail experts
computer tools	Better reliability for the service	less e-mail "lost down a black hole"

Figure 3: User Benefits

Figure 3 summarizes a number of concrete examples of the benefits to end-users obtained from some of the activities which are coordinated by COSINE MHS.

4.5 COSINE MHS Service Goals

The service's goals are to be achieved in a phased approach over a period terminating on 31 December 1992, after which the service is to become self-sustaining. The current minimum service goals are as follows; additions in the light of experience may be made over time as the service evolves.

- to enable national end-users to use an international X.400(84) MHS;
- to provide the national MHS managers with information on other national MHS services participating in COSINE MHS;
- to provide national MHS managers with an error reporting service on the international level;
- to link the COSINE community to similar communities in other parts of the world;
- to provide an international service over national and local services;
- to encourage the participation of other international MHS initiatives in the COSINE MHS service;
- to encourage the public X.400 services to provide a better service than COSINE MHS;
- to provide tools for maintaining and managing the COSINE MHS service;
- to ensure full connectivity to existing international RFC mail networks;
- to encourage the migration of experimental MHS services to becoming fully operational;
- to provide a single, unified European MHS service;
- to provide information to the CPMU so that an on-going, self-sustaining service can be maintained beyond the period of the COSINE subsidy.

The analysis of how to achieve the service goals led to the identification of seven main activities as being critical for the successful completion of the project. These are indicated in Figure 4 which shows the high-level service model employed by the project team in the running of the service.

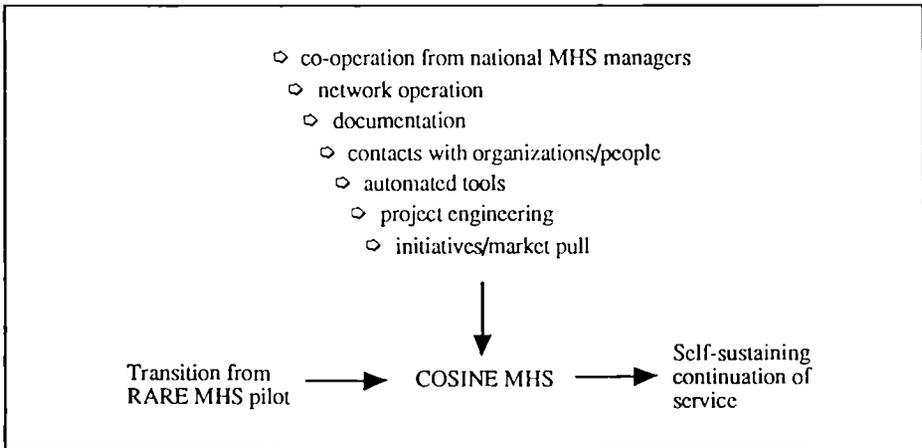


Figure 4: Model of the COSINE MHS Service

4.6 Relationships to Other Networks and Activities

A number of technology based messaging networks exist, for example: UNIX based mail (SMTP, UUCP), IBM based mail and even fax or bulletin boards. COSINE MHS assists in the co-operation between the world-wide X.400 MHS research network and the non-X.400 networks to coordinate address mapping tables used by gateways to the networks. A separate project team operates the COSINE MHS gateway service for the research community to connect to other mail networks (for example: High Energy Physics NETWORK (HEPNET), Space Physics Astronomy Network (SPAN), BITNET, European Academic and Research Network (EARN) and the Internet). The two project

teams work to produce a seamless, unified COSINE electronic mail service and use a common contact point for inquiries from end-users.

The service co-operates with a number of other COSINE activities, including:

- PARADISE X.500 directory service: inquiries to this directory allow easy retrieval of a user's electronic mail address.
- CONCISE user information service: access to this service can be made via X.400 electronic mail, and the service provides a help desk aimed at the non-computer specialist users.

COSINE MHS is complementary to the CEC's Y-Net project which has as its emphasis researchers who do not have easy access to a national research network.

All these factors, plus the use of X.400 as the basis for inter-connection to public electronic mail service providers, contribute to the project teams' ability to provide a powerful connectivity solution encompassing the entire research community and non-research users.

4.7 Future Plans

The project team monitors the development of new functional additions in the MHS area and will encourage their deployment in the research community when tangible extra benefits for end-users are detected. Some sort of multi-media messaging, probably based upon Office Document Architecture (ODA) standards, seems especially promising in the near term. Migration of the service to the higher functionality provided by X.400(88) has already started, although suitable migration strategies for the whole service still need to be generated.

Possibilities for increasing connectivity to new MHS networks (for example in East Europe) are emerging and these new networks will continue to be assisted by the service.

5. References

[1] S.E.Kille, University College London. Handling QOS (Quality of service) in the Directory, March 1991. Internet Draft.

Y-NET THE ESPRIT PAN-EUROPEAN COMMUNITY OSI NETWORK

C. BERRINO and D. MANUELLO
Y-NET Management Unit
Teleo S.p.a.
Rue du Trone 12
B-1050 Brussels

SUMMARY

The main purpose of Y-NET, Esprit project 5700, is to establish a Pan-European distributed OSI service infrastructure to ensure that all the participants in ESPRIT and other European Community R&D programmes will enjoy improved communications and exchange of data, based on OSI conformant systems offered by the Esprit Manufacturers (i.e. Bull, Olivetti and Siemens-Nixdorf Informationssysteme). One of the major features of Y-NET project is the management structure, that is based on two organizational levels including a central co-ordination and management unit, **Y-NET Management Unit (YMU) responsible for the whole project**, that the CEC contracted to TELEO S.p.A. (an Italian company belonging to IRI/STET group) and a set of **National Operational Units (NOUs)** at national level responsible for operating of Y-NET OSI nodes, administering and supporting the Users and promoting Y-NET in their countries. The NOUs are sub-contracted from YMU to National organizations. In the paper, a detailed technical presentation is provided.

1. INTRODUCTION.

As Europe moves towards the completion of the Unique Market in 1992, all the obstacles to the free movements of the Resources are being removed. The key of the success in this environment is the **INFORMATION** that must be correct, up-to-date, accurate, at the right time and fully available in a short time to everybody.

In order to satisfy all these requirements, the **INFORMATION TECHNOLOGY (IT)** can be considered the Solution; in fact Industries, Governments, Businesses and Educational Bodies are investing a lot of money for Research & Development in IT.

Since the 1984, the European Community (EC) launched the ESPRIT Programme in order to push the IT R&D in Europe, in liaison with all the interested Bodies and partially founded by the EC itself.

In order to success in the ESPRIT programme, EC has considered as mandatory requirement to make available sophisticated electronic network services to the Researchers, belonging to different European countries but involved in the ESPRIT projects, to allow them to exchange Information.

These services are known as Information Exchange Services (IES), and Y-NET is one of them.

The Y-NET initiative is based upon three complementary requirements:

- Communication needs within ESPRIT and other CEC R&D programmes;
- Exploitation and usage of developments from ESPRIT projects related to Open System Interconnection (OSI);

- Need for an organisational framework for end-user oriented support for participants of ESPRIT and other CEC R&D programmes.

The communication needs exist not only among Researchers, as stated before, but also between project participants and the staff of the EC (project reports, project control and administrative information).

On the other hand, for ESPRIT, the progress of OSI standardisation and its applications has always been of high importance, in fact a large number of cooperative projects, related to OSI, have been sponsored in the last years.

In this context and initiated through discussions in the ESPRIT Advisory Board (EAB), Manufacturers from ESPRIT (i.e. Bull, Olivetti and Siemens-Nixdorf Informationssysteme) took an initiative in cooperation with the IES to make the results of their work available to the European R&D community in order to provide pilot OSI services.

In this sense, we can define **Y-NET** as the ESPRIT pilot OSI project that has been launched to establish a Pan-European distributed OSI service infrastructure to ensure that all the participants in ESPRIT and other European Community R&D programmes will enjoy improved communications and exchange of data, using OSI conformant systems.

In order to present an overview of the project, it is possible to identify the following items:

- The Technical Approach;
- The Organizational Approach;
- The User Community;
- The OSI services offered.

1.1. TECHNICAL APPROACH

The basic technical approach of Y-NET comprises the establishment of one or more OSI nodes (called **Service Points**) in each member state each of which provides a set of OSI services.

The hardware and software, including maintenance for the initial European-wide Y-NET configuration is provided by the European Manufacturers free of charge. The set of inter-connected heterogeneous systems will commonly provide the Y-NET service.

All the equipment provided by the Manufacturers is conform to the European profiles. Moreover, a number of interoperability tests between systems in Y-NET have been specified and are performed periodically in order to guarantee the network interworking.

Then, the Y-NET network is based on the following elements:

- 11 nodes located in the Member Countries of the CEC (there is one node for Belgium and Luxemburg): these nodes, that are called Service Points, are full OSI systems which provide User Interface services to end-users and OSI services to users supplied of OSI equipment;
- a special node (located in France) offering a gateway service between X.400 (OSI standard E-Mail) and EUNET (the European-wide UNIX mail network based on Send Mail Transfer Protocol - SMTP)

A more precise description concerning both the Architectural and the Testing Aspects is presented in chapter 3 and 4 respectively.

1.2. ORGANIZATIONAL APPROACH

One of the major features of Y-NET project is the management structure, that is based on two organizational levels:

- a central co-ordination and management unit, **Y-NET Management Unit (YMU) responsible for the whole project**, that the CEC contracted to TELEO S.p.A. (an Italian company belonging to IRI/STET group);
- a set of **National Operational Units (NOUs)** at national level responsible for operating of Y-NET OSI nodes, administering and supporting the Users and promoting Y-NET in their countries. The NOUs are sub-contracted from YMU to National organizations.

The advantages of such organisation is to avoid to create a big centralized management unit for all the network and to provide a national support to the users using the national language for documentation, Help Desk service, and so on.

1.3. Y-NET USER COMMUNITY

As stated before, the access to Y-NET SPs will be allowed to all participants in **Esprit** and other **European Community R&D programmes**, in particular to the people that work in the Industrial R&D environment. Y-NET is oriented towards two categories of Users:

- Researchers who, within their organisations do not have access to OSI services today. They can use Y-NET via direct connection to their SPs. This service is particularly oriented towards **Small and Medium Enterprises (SMEs)**.
- Researchers belonging to industrial organization that have already set-up OSI domains should consider using Y-NET to achieve connections with their partners in research projects.

1.4. OSI SERVICES OFFERED

Each Service Point provides the following OSI service:

- X.400 Electronic Mail - Message Handling System (MHS)
- File Transfer and Access Management (FTAM)

In the future, as soon as other services will be available from the Manufacturers (i.e. X.500 Directory Service, Office Document Architecture (ODA), Electronic Data Interchange (EDI) and so on), they will be introduced inside the Y-NET network.

In this paper we would like to present the technical approach that we have followed to set up the first of the OSI services that Y-NET offers to its customer:

The Message Handling System (MHS) or X.400 Electronic Mail.

In the next paragraph we describes the major characteristic of the X.400 service (chapter 2), the network architecture (chapter 3) and the testing strategy adopted to assure the conformance and the interworking of the different systems supplied by the Manufacturers (chapter 4).

2. MHS SERVICE DEFINITION

One of the major target of the Y-NET project is to merge the advantages of a distributed network managed by national companies (i.e. 11 Service Points - SPs located in all the EC countries) and those of offering to the user community the same Level of Service (LoS) in each SP.

Considering the two level of the project management organisation, it seems obvious that YMU is responsible of the Level of Service definition for the whole network. In defining that, YMU has taken in consideration these important issues:

- Y-NET X.400 network is a private network, where all the nodes are Private Domains (PRMDs) that shall conform to the European profile ENV41201 (that concerns the MTAs acting as Private Domains). This implies that all the service requirements defined by the profile shall be included in the Y-NET LoS.
- all the systems provided offer to the user a set of capabilities not included in the above mentioned ENV profile, but considered very useful from the user point of view. These implies that these additional capabilities shall be included in the Y-NET LoS.

On the base of these issues, YMU has defined the initial level of the Y-NET X.400 Service Profile that should be considered as the **Kernel** of the X.400 service offered by Y-NET. It includes:

- the standard service elements indicated as mandatory in the ENV41201 and the optional ones but supported by all the systems;
- the common set of non-standard service capabilities offered by all the systems;

that are described in the next two paragraphs.

2.1. STANDARD SERVICE ELEMENTS

This section contains the list of standard service elements that each X.400 system providing the Y-NET service shall offer at the User Interface (UI).

Each Service Element can be either **Supported** or **Non-supported** by the equipment.

Non-supported means that the service element is not made available to the service user. *Supported* means that the service element is available to user at the UI level.

Moreover, there is a different meaning associated to a service supported in origination and in reception:

- *Supported in origination* means that the X.400 system offer to users the capability to generate the request of that service at the UI level;
- *Supported in reception* means that the X.400 system offers the capability to present to users, at the UI level, the service indication received.

As stated in the ENV41201, the fact that a service element is supported in origination does not imply that the same element is also supported in reception and vice versa.

Anyway, YMU decides to include in the LoS the set of Service Elements supported either in reception or in origination. The table reported in the following reflect this kind of difference.

STANDARD
SERVICE ELEMENTS

ORIGINATION RECEPTION

1 IP-message Identification	S	S
2 Typed Body	S	S
3 Blind Copy Recipient Indication	N	S
4 Non-receipt Notification	N	N
5 Receipt Notification	N	N
6 Auto-forwarded Indication	N	S
7 Message Circulation	N	N
8 Originator Indication	S	S
9 Authorizing Users Indication	N	S
10 Primary and Copy Recipients Indication	S	S
11 Expiry Date Indication	N	S
12 Cross-referencing Indication	N	S
13 Importance Indication	S	S
14 Obsolete Indication	N	S
15 Sensitivity Indication	S	S
16 Subject Indication	S	S
17 Replying IP-message Indication	N	S
18 Reply Request Indication	S	S
19 Forwarded IP-message Indication	N	S
20 Body Part Encryption Indication	N	S
21 Multi-part Body	S	S
22 Timed Obsolete Indication	N	N
23 Content Type Indication	S	S
24 Converted Indication	NA	S
25 Delivery Time Stamp Indication	NA	S
26 Message Identification	S	S
27 Non-delivery Notification	S	NA
28 Original Encoded Information Types Indication	S	S
29 Submission Time Stamp Indication	S	S
30 Delivery Notification	S	NA
31 Disclosure of the Recipients	N	S
32 Grade of Delivery Selection	S	S
33 Multi-destination Delivery	S	NA
34 Conversion Prohibition	S	S

Anyway, it is important to explain the meaning associated to the Service Element "Multi-part Body", because it depends on the context of the X.400 network. The Y-NET X.400 Network allows to transfer two different kinds of body parts:

- an ASCII text as a IA5TEXT body part;
- a BINARY file in transparent mode as an UNIDENTIFIED body part.

The second capability is considered very important because, compared with other E-Mail services that are currently used in R&D environment, offers to the Y-NET users the opportunity of transferring, among them, through X.400 Network any kind of file prepared in advance using a Word Processor (i.e. Microsoft Word files), a Graphic tool (i.e. Power Point files) or any other software tool.

2.2. ADDITIONAL SERVICES

All the X.400 systems offer the following non-standard services to the users:

- using of **alias** names, defined either by the user or by the System Administrator, instead of the O/RNames;
- using of **distribution lists**, defined either by the user or by the System Administrator, to make easier the use of the multi-destination delivery service.
- storing both incoming and outgoing messages in the mailbox.
- using Kermit protocol to transfer to the mailbox storage the files prepared in the remote workstation and vice versa
- browsing facilities to read messages.

These additional services are considered very interesting and important, because contribute in making easier the access to the service.

3. Y-NET X.400 NETWORK ARCHITECTURE.

The design of the X.400 network architecture concerns two main aspects related to each other:

- naming: the structure of the O/R name of the network users (to make them unambiguous inside a Management Domain).
- routing: the strategy that should be applied to route a message from the origination to the destination point.

The link between these aspects is quite evident: in fact, the routing of a message can be done using the information contained in the address (i.e. the P1 O/R name) of the recipient user.

It has been YMU aim to maintain the naming structure as independent as possible from the routing because:

- naming structures should be stable;
- routing could change dynamically depending on the traffic and on the availability of resources

In the following sections the naming structure and the routing policy of the Y-NET X.400 network are defined based on equipment provided by the Manufacturers.

3.1. Y-NET FEATURES

Y-NET X.400 network is characterized by the following features and requirements:

1. It links X.400 MTA, classified as PRMD and situated in different countries. **Y-NET is a pan-European network.**
2. Y-NET service users are connected and are able to exchange messages with users of other OSI and non-OSI Research Networks. **Y-NET is an "open" network.**

To satisfy these requirements the Y-NET X.400 network topology is based on the following elements:

- 11 nodes located in 11 countries of the CEC: these are called Service Points (SP) and are operated by National Operational Units (NOU).
Each SP is a full X.400 system and has a twofold role:
 1. it offers the service to end-users giving the access to a User Interface;

2it is able to route messages.

- A gateway between X.400 and EUNET.

Before proceeding it is important to give a classification of the possible users of this network:

1. "**direct**" users: the users that access to the Service Point from a remote PC;
2. "**indirect**" users: the users that access to the Service Point from their private MTA.

3.2. NAMING STRUCTURE OF Y-NET DIRECT USERS.

Aim of this paragraph is to describe the schema that is followed to assign O/RNames to Y-NET direct-users. In fact it is supposed that indirect users have already got their O/RNames and that these are accepted whenever their O/RName are complying to the schemas defined in the X.400 Recommendation.

Because each SP is a full X.400 system, the O/RName structure adopted by YMU is the FORM1/VARIANT1 defined in the X.400 '84 Recommendation.

Each O/R Name includes the following information:

COUNTRY NAME: the name of the country in which the SP is located (interpreted as ISO 3166 ALPHA 2 country codes).

ADMINISTRATIVE DOMAIN NAME: in countries where at least one ADMD name exists or is registered, this name should be included in the ADMD field. Otherwise, in accordance with what is stated in the European Standard ENV 41201, the content of this field should be a single space.

PRIVATE DOMAIN NAME: in each country, it should be always **Y-NET**.

ORGANIOZATION NAME: to each MTA in the private domain Y-NET should be assigned the value SP followed by an increasing number. The first MTA will be assigned the value SP1.

PERSONAL NAME: it is the name of the user.

Summarizing what is said above a typical Y-NET O/R Names is:

CN		BE
ADMD		RTT
PRMD		Y-NET
ON		SP1
PN	SN	Speth
	GN	Rolf

The most important consequence of this O/RName structure is that the ADMD field is filled with the name of the Public Service Providers. This implies that:

- Y-NET don't have any problem in the interconnection with the Well Known Entry Points WEPs, because we are on line with the naming structure accepted by Rare-Cosine X.400 profile.
- Y-NET is ready to be interconnected with public X.400 Networks.

3.3. ROUTING Policy for y-net SERVICE POINTS

Y-NET X.400 network is made of a set of distributed nodes that are able to offer the access to OSI services and to route messages to other MTAs belonging either to Y-NET or to other X.400 networks.

As a consequence, the interconnections that the SPs have during the starting phase of the project are based on a PRMD to PRMD relationship, in terms of European profiles it means that our MTA should comply with ENV41201 standard profile.

In the following two aspects of routing are considered:

A) ROUTING INSIDE Y-NET X.400 NETWORK

Regarding the routing inside Y-NET X.400 network, each SP user is able to exchange messages with all the Y-NET community, then each SP has to know all the others.

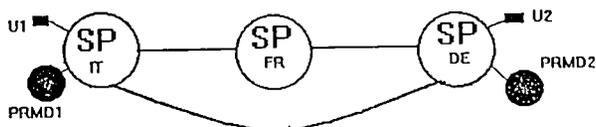


Figure 1 - Routing inside Y-NET network

B) ROUTING WITH OTHER OSI NETWORKS

One of the major aim of Y-NET is to allow the communication of our users with other R&D networks, especially with the Rare-Cosine network users.

In order to make it possible, YMU in agreement with Cosine S2.1 MHS Service management has decided to adopt the following strategy:

in each country the Service Point is connected to the Well Known Entry Point (WEP) of this country (or one of the WEPs, if there are more than one).

Besides, each network should try to route the message as much as possible close to the destination node of the other network, it means that the exchange of messages between the two X.400 networks should happen as follows:

national traffic

when Y-NET users submit to the SP a message for a Rare-Cosine user, then the SP routes it to the national WEP known. It is a WEP task to deliver it to the recipient. Traffic from Rare-Cosine to Y-NET should follow the same path.



Figure 2 - Interconnection between Y-NET and Cosine MHS network (national level)

international traffic

when Y-NET users submit a message for a Rare-Cosine user, the originating SP routes it to the SP sited in the country of the recipient; in this way the message arrives in the recipient country staying as much as possible in Y-NET. Then this SP should route it to its WEP, that shall deliver it to the recipient.

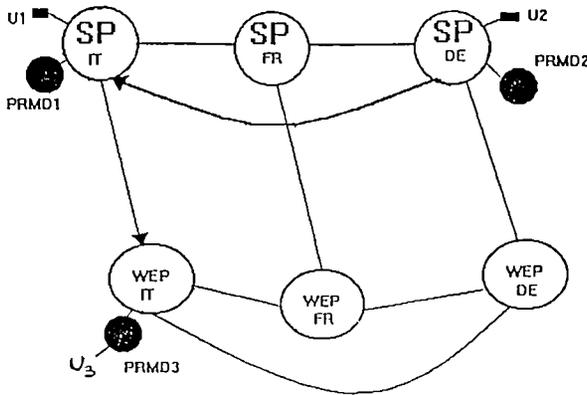


Figure 3 - Message routing from Y-NET SP towards Cosine WEP (international traffic)

when Rare-Cosine users submit a message for a Y-NET user, then the message, following internal routing rules, is addressed to the SP of the recipient country: again the message stays as much as possible inside the Rare-Cosine network. Afterwards, the SP shall deliver it to the recipient.

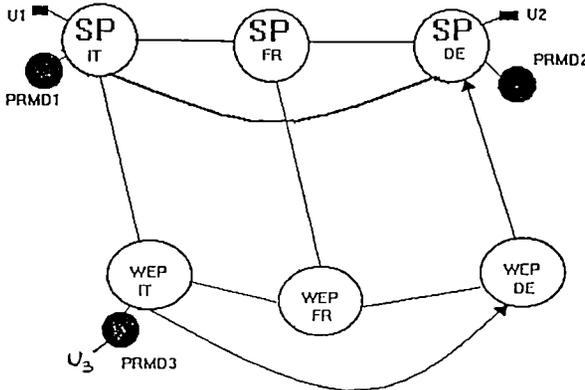


Figure 4 - Message routing from Cosine WEP towards Y-NET SP (international traffic)

3.4. INTERCONNECTION WITH NON-OSI NETWORKS

One of the objective of Y-NET project is provide interconnectivity between OSI and non-OSI E-Mail systems, for this reason:

- it has been included in the network a special Service Point (located in France) offering the gateway services towards EUNET, that is the European-wide Unix network.

- Y-NET is going to make use of a EUROKOM system (located in Ireland), offering gateway services between X.400 and Eurokom E-mail system.

In order to make possible the mail exchange between Y-NET and EUNET users YMU considered these main issues:

Definition of RFC-822 naming schema for Y-NET users

In order to allow the EUNET users to exchange mails with Y-NET users, it has been defined a translation rule that should be applied to map the X.400 O/Rname on the RFC-822 address.

Then, whenever EUNET users want to send a mail to Y-NET users they just have to apply the translation rule defined and create the corresponding RFC-822 address associated to each of our users.

The EUNET network is responsible to route the message towards the Y-NET Gateway, that is responsible to map it in the exact X.400 O/Rname format and send the message towards the right Y-NET Service Point.

Use of X.400 naming schema to address EUNET users.

YMU should assure Y-NET users to have the possibility of sending messages to EUNET users using X.400 systems.

This means that we had to find a common rule to insert in the X.400 address schema the RFC-822 name in order to allow a Y-NET user to address an EUNET one just using the X.400 User Interface available at the Service Point site for the direct users, and the proprietary X.400 User Interface for the indirect users.

On the other hand, the X.400 address should be also used to address the unique Y-NET Gateway node located in France.

Consequently, YMU defined the following mapping strategy:

Use of the CN, ADMD and PRMD names in order to address the Gateway:

CN	=	FR
ADMD	=	ATLAS
PRMD	=	YNETGW

Use of the DomainDefinedAttribute (DDA) List to map the EUnet address, when the originator is a Direct User or a an Indirect one who has an X.400 system supporting DDA.

Use ON and OUs fields, when Indirect Users equipment does not support the DDA.

For example, considering this RFC-822 name:

medina @ ac.ipc.es

can be mapped as follows:

case a) Direct User sends a message to EUnet User

CN	=	FR
ADMD	=	ATLAS
PRMD	=	YNETGW
dda.type	=	RFC-822
dda.value	=	medina @ ac.upc.es

case b) Indirect User

CN	=	FR
ADMD	=	ATLAS
PRMD	=	YNETGW
ON	=	es
OU1	=	upc
OU2	=	ac
Surname	=	medina

YMU suggests to use the dda, whenever is possible, (i.e. the direct users should adopt this solution) and otherwise the sets of fields ON, OUs, personal name (i.e. the indirect users should adopt this solution if their X.400 systems do not offer the dda capability)

4. Y-NET TESTING ASPECTS: A QUALITY ASSURANCE

Testing of the equipment used by the Service Points is considered as an important activity to be monitored and undertaken, because one of the most important commitment for Y-NET is to offer an OSI network in which all the heterogeneous Systems involved are both standard compliant and inter-operable.

To satisfy these requirements, a number of testing activities has been performed in order to cover the following aspects:

- **Conformance Testing:** to verify that the Manufacturer's equipment is conformant to the requirements of the Y-NET Project (i.e. the compliance with the European Profiles);
- **Interoperability Testing:** to verify that each Service Point is able to interconnect with the remaining Service Points and with other systems not belonging to Y-NET;

4.1. CONFORMANCE TESTING

Referring to the commitment stated above (i.e the need to have in Y-NET only OSI **standard** systems), it has been considered of extreme importance as assurance of quality the production of a conformance test report provided by one of the European Accredited Laboratories in order to prove the compliance of these equipment to the European profiles.

Then, Y-NET Management Unit (YMU) has coordinated with the Manufacturers the Conformance Testing of the equipment they are providing to Y-NET project. Concerning X.400 E-Mail, YMU requested the Manufacturers to supply for the first Pilot Phase (April - May 1991) an **unofficial** Test Report provided by their own Test Laboratories, and then to supply, before the network becomes operational (September 1991) an **official** Test Report provided by an Accredited Laboratory.

4.2. INTEROPERABILITY TESTING

As it is underlined above, the Conformance Testing is very important to guarantee the quality of the equipment, nevertheless, considering the operational point of view, the Interoperability Testing assumes the same or more importance. In fact, it is the only way to guarantee the provision of a network able to interwork.

Considering the importance of this aspect, both the Manufacturers and the YMU, before of the equipment delivery to the Service Point sites, have performed an Testing Campaign based on a number of tests defined in the EUROSINET Interoperability Guide.

In any case, YMU, being responsible to guarantee the interworking of the Y-NET network and define the Y-NET Interoperability policy, has defined a set of Interoperability Tests in the "Y-NET Interoperability Tests Guide". These have been designed to demonstrate at least the following:

- the correct implementation of services available at the User Agent level both when the Y-NET user is originator and when he is recipient;
- the transfer of files between the Service Point and remote workstations using the KERMIT transfer protocol
- the correct relaying of X.400 messages offered by each Service Point to the other SPs, the gateways and the other Private Domains connected to Y-NET.

These tests are performed by the personnel operating the Service points (i.e. the NOUs) and then the results of the test run are provided to the YMU through an Interoperability Test Report.

NOU personnel should run Interoperability Tests in the following events:

1. when the OSI equipments (i.e. X.400 E-Mail and FTAM) are installed at the Service Point site.

In order to accept the equipment the NOU Personnel has to verify that all the documented facilities of the service are able to work correctly. Then, the NOU personnel should run a few interoperability tests with the other active SPs before accepting this equipment.

2. when private OSI systems request to be registered to Y-NET service.

Before accepting an indirect user it must be verified the ability of the user system to interoperate with the Y-NET SP;

3. when private X.400 network request either to be registered to Y-NET service or to interwork with Y-NET (e.g. RARE/COSINE MHS network).

Before activating the interconnection with the other research network, it must be verified that the two networks can interwork and route correctly.

5. CONCLUSION.

As we widely explained in the paper, **Y-NET** is the ESPRIT pilot OSI project that has been launched to establish a Pan-European distributed OSI service infrastructure to ensure that all the participants in ESPRIT and other European Community R&D programmes can enjoy improved communications and exchange of data, using OSI conformant systems.

Until now, YMU is able to offer the Message Handling System service and has planned to introduce the File Transfer part of FTAM service by January 1992. Anyway, the introduction of other services is under YMU consideration (i.e. X.500 Directory Service, ODA, EDI and so on). This will be possible as soon as the Manufacturers make their products available.

It is now important to point out the main advantages of the Y-NET initiative, that we can be summarized in the following items:

- Use of OSI commercial systems provided by the major European Manufacturers that assure their strong support to Y-NET initiative;
- Use of OSI systems that comply to the European Standards;
- Centralized responsibility of the Y-NET service provision: this is provided by the Y-NET Management Unit located in Brussels;
- Operation of the Y-NET service offered on a national basis in each member state. This implies that the National Operational Units offer the User Support, concerning with Help Desk, Hot Line, User Guides, to the Y-NET users in their National Languages;
- Full interconnection with the other Research and Development Networks (i.e. Rare-Cosine, EUNET).
- Possibility to exchange formatted files (e.g. files written using PCs word processor).

BASIC RESEARCH

BUILDING USER INTERFACES: ORGANIZING SOFTWARE AGENTS

L. NIGAY, J. COUTAZ

Laboratoire de Génie Informatique (IMAG)

BP 53 X, 38041 Grenoble Cedex, France

fax: (33) 76 44 66 75, tel. (33) 76 51 48 54

email: nigay@imag.fr, joelle@imag.fr

SUMMARY

This article presents the results of the system architecture modelling activity performed in the context of the Esprit BRA 3066 AMODEUS project. The goal of this research package is to provide insights and effective guidelines for designing and implementing interactive software. We propose to represent an interactive system at different levels of granularity ranging from a revisited version of the Seeheim model up to a highly parallel multi-agent PAC organization. Although multi-agent models are considered to be a promising way of supporting human factors as well as software engineering requirements, they are hard to apply. This article provides a set of heuristic rules that support the software design task.

1. Introduction

The definition of an appropriate software architecture is an important issue in user interface design. Without an adequate architectural framework, the construction of interactive systems is hard to achieve, the resulting software is difficult to maintain and iterative refinement is made impossible. Software tools, such as toolkits, application skeletons, and UIMS's, designed to support the construction of user interfaces, do not alleviate the problem.

Toolkits such as Motif (1), application skeletons such as MacApp (2), and presentation-driven UIMS such as Interface Builder (3), do not embed the architectural principles that a software designer needs. For example, the "callback procedure" paradigm made popular by X Window (4), does not make any distinction between "semantic-domain dependent" concepts and "user interface-presentation dependent" issues. Thus, without an adequate software framework, the resulting interactive system may be an incredible mixture of concerns. Software tools for the construction of user interfaces will not solve architectural problems as long as the construction of these interfaces requires programming. Clearly, a reference model for identifying and organizing the components of interactive software is still a necessity.

Since Seeheim (5), a number of software architecture models have been devised and discussed in multiple workshops and working groups (6, 7). We observed at one special interest group on UIMS's at the CHI'91 conference, that all of these models are too general to be useful; they are too ambiguous, thus leading to multiple possibly wrong interpretations. One such misinterpretation is the sequential linguistic approach mapped onto the Seeheim model (8). Other approaches such as PAC (9) and MVC (10), put the emphasis on highly parallel architectures organized in terms of cooperative agents.

Multi-agent models offer some promising foundations for software engineering (e.g. modularity and distribution). As important, they support the opportunistic behavior of the end-user as well as multiple parallel inputs and outputs (9). However, our own experience with PAC revealed that, given the external specifications of a particular system, software designers have trouble identifying the agents necessary to the implementation of that system. This article aims to provide a preliminary cookbook for defining such agents.

This work has been performed in the context of AMODEUS, Esprit BRA project no 3066. Section 2 summarizes the goals of AMODEUS and indicates how our action fits in this general framework. In section 3, we provide a general description of the architecture of an interactive system and indicate the key components of that architecture. The central component is then refined in terms of agents. The structure of these agents as well as guidelines for defining their inter-relationships are described in sections 4 and 5 respectively.

2. The Context: the AMODEUS project

The AMODEUS project investigates how the various kinds of modelling techniques in HCI for representing the user, the system, and the design space, can be integrated into a unified framework (11). As a means of unification, the scenario methodology has been adopted (12, 13): a set of scenarios drawn from real experiments are analyzed by the different modelling approaches. The outcome of the exercise provides a basis for understanding the strengths and weaknesses of each modelling approach, their interrelationships and patterns of complementarity.

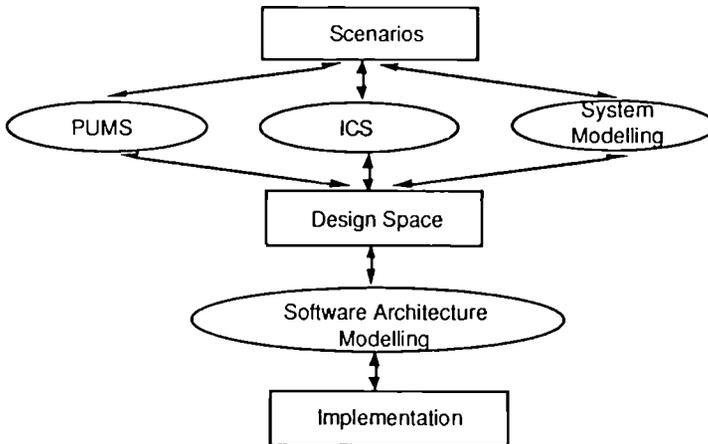


Fig. 1: The modelling techniques of the AMODEUS project.

As shown in Figure 1, a scenario is used as "tool for thought". It is an instance of a pertinent class of problems expressed in natural language. For example, a scenario may propose two design alternatives such as fish-eye views or the scrolling mechanism for displaying information on the screen. The system modelling technique expresses the problem in a formal way (14). Based on an unambiguous expression of the problem, user modelling techniques such as ICS (15) and PUMS (16) make predictions about user's behaviour and warn the designer of usability problems. The formal description of the problem as well as predictions and warnings inform the design space. Software

architecture modelling (our approach) starts with the design solutions selected in the design space and leads to the implementation of these solutions. Like the other modelling approaches, it informs the design space in terms of specific criteria. Salient criteria at this level might include efficiency and reusability, for example.

As shown in Figure 2, software architecture modelling activity in AMODEUS is a two step process:

- the first step is based on heuristics reasoning: a software architecture is derived for a particular design solution which, in turn, is drawn from psychological knowledge in HCI. This architecture makes explicit the agents involved in the software solution;
- formal reasoning is then used to describe the behaviour of the agents as well as their relationships in a precise and unambiguous way. Such description leads the way to the automatic generation of user interfaces.

In this article, we are concerned with the heuristic approach only. The formal notation, which combines CSP and Z descriptions, can be found in (17).

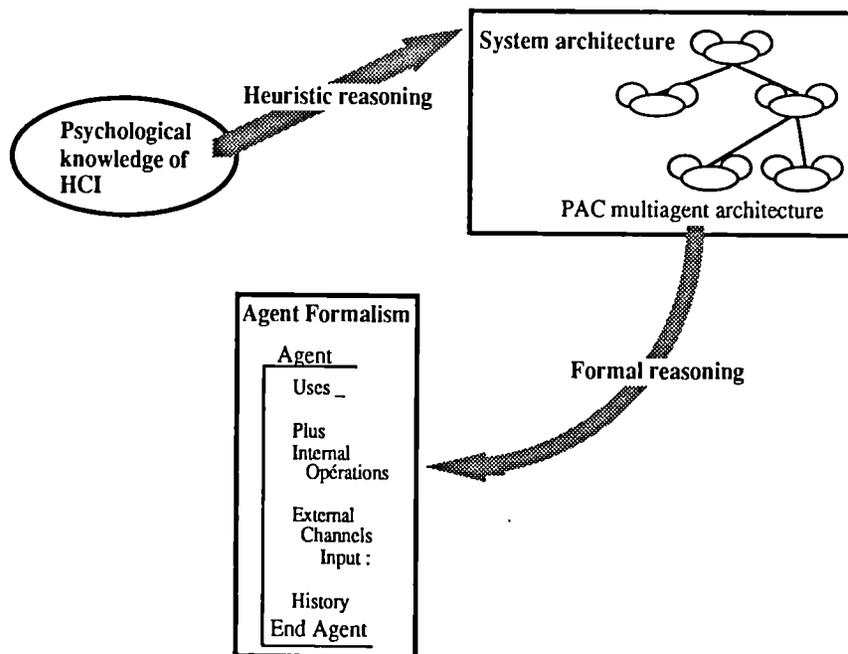


Fig. 2: Heuristic and formal reasoning used in AMODEUS for designing user interface architectures.

3. Overall Organization of an Interactive System

There is a consensus in user interface software design that there is an important distinction between the functional core and the user interface. The functional core of an interactive system is the software portion which implements domain dependent concepts. The user interface denotes the presentation (i.e. the *image* as defined by D. Norman in (18)). This distinction has been admitted as a sound basis for iterative refinement and helped in the emergence of the UIMS technology.

The Seeheim model refined the user interface portion of an interactive system into three components: the interface with the functional core, the dialogue controller and the presentation component. The role of each component was roughly described as the semantic, syntactic and lexical functionalities of the user interface. As mentioned above, the imprecision of this definition opened the way to multiple interpretations.

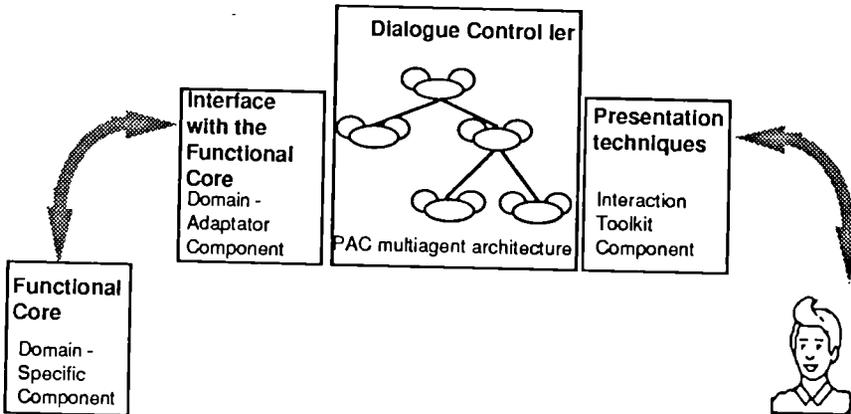


Fig. 3: The components of an interactive system.

Figure 3 shows our view of a general model of an interactive system. One salient property of this model is a nice balance organized around a key component: the dialogue controller. At the two extremes, the functional core and the user play a symmetric role at a high level of abstraction (the domain concepts). Next, come the two interface components: one for the functional core, a second one for the interaction toolkit. These components are discussed further in the following paragraphs.

The User and the Functional Core

The user and the functional core both produce and consume information (e.g. events) through the dialogue controller. This symmetric view of the general functioning of an interactive system where the key component is equally driven by the functional core and the user does not impose any model on the control of the interaction. The question of external VS internal VS mixed control (19) is irrelevant at this level of description. Clearly, the choice between the alternatives is guided by the case at hand.

This balance is in contrast with earlier interpretations of the Seeheim model which imposed an *a priori* scheme on the control of the interaction. First generation UIMS's, inspired by the compiler technology, found it very convenient to view the functional core as a semantic server. This external control of the interaction was adequate for domains such as graphics and text editing; it was inappropriate for control process systems where the functional core may produce information in an asynchronous manner from the point of view of the user (e.g., alert messages in a nuclear station security system).

Interface with the Functional Core

The interface with the functional core (IFC) serves as a buffer between the dialogue component and the domain specific concepts implemented in the functional core. It is designed to absorb the effects of change in its direct neighbours. As does any boundary, it implements a protocol. A protocol is characterized by temporal strategies, by the nature of data exchanged and by some linking mechanisms. These three dimensions are discussed in (20).

Data exchanged need further explanation.

The IFC is a good place for semantic enhancement and semantic delegation. Both of these design operations can be expressed in terms of conceptual objects. (In our discussion, the term "object" covers the encapsulation of a set of data and operators.) A conceptual object is supposed to match the mental representation that the user elaborates about a particular domain concept. It may be the case that the functional core, driven by software or hardware constraints, implements a domain concept in a way that is not adequate for the user. For example, the concept may be split into multiple data structures.

Semantic enhancement may be performed in the IFC by defining a conceptual object which reorganizes the data of the functional core. Reorganizing may take the form of glueing together data structures of the functional core into a single conceptual object or, conversely, splitting a concept into multiple conceptual objects. Semantic enhancement may also take the form of a conceptual object that extends the properties of a domain concept. For example, the "status-man" described in (21), which counts the occurrences of semantic errors according to their level of gravity, extends the notion of error. It provides the user with a synthetic view of the semantic validity of the entity being built with the system. Counting errors is certainly not relevant to the functional core although it provides the user with useful task dependent information. A conceptual object is a good way to solve the problem.

Semantic delegation may be performed in the IFC to enhance performance. The semantic quality of feedback may require frequent round trips to the functional core. This long chain of data transfer may be costly with respect to time. Therefore response time may be inconsistent with the expectation of the user. Semantic delegation, which consists of down-loading functional core knowledge into the user interface is a way to reduce transmission load. In particular, if the functional core is implemented as a distinct process running on a distinct processor, it may be judicious to use the IFC as a cache of the data maintained in the functional core. As described in the next section, semantic delegation may be performed in the dialogue controller as well.

The Presentation Techniques

The presentation techniques component (PTC) implements the physical interaction with the user via hardware and software. It includes a set of interaction techniques that defines the *image* of the system (output). It also deals with the events produced by the user (input). If we consider current practices, presentation techniques are constructed from concepts made available in user interface toolkits.

The PTC is the location for handling low level multi-media events (for input as well as for output). For example, low level input events produced by voice and gesture input devices may be time-stamped and queued in the same way mouse events and

keyboard events are currently processed by windowing systems. Multi-media events should be combined into higher level events in the same way as keyboard and mouse events are currently combined into higher abstractions by toolkit dialogue boxes.

The Dialogue Controller

The dialogue controller is the key component of our model. Contrary to a number of user interface models, the dialogue controller is not a monolithic "obscure potato"! Instead, it is organized as a set of cooperating agents which smoothly bridges the gap between the IFC and the PTC. The nature of these agents and their relationships will be further described in the next sections. For now, we need to justify the overall functionality of the dialogue controller.

The overall functionality

The dialogue controller has the responsibility for task-level sequencing. Each task or goal of the user corresponds to a thread of dialogue. This observation suggests the choice of a multi-agent model. Indeed, a multi-agent model distributes the state of the interaction among a collection of cooperating units. Modularity, parallelism, and distribution are convenient mechanisms for supporting multi-thread dialogues. One agent or a collection of cooperating agents can be associated to each thread of the user's activity. Since each agent is able to maintain its own state, it is possible for the user (or the functional core) to suspend and resume any thread at will.

The dialogue controller receives events both from the functional core via the IFC, and from the user via the PTC. Bridging the gap between an IFC and a PTC has a number of consequences. In addition to task sequencing, the dialogue controller must perform data transformation and data mapping:

- 1) An IFC and a PTC use different formalisms. One is driven by the computational considerations of the functional core, the other is toolkit/media dependent. In order to match the two formalisms, data must be transformed inside the dialogue controller.
- 2) State changes in the IFC must be reflected in the PTC (and vice versa). Therefore links must be maintained between conceptual objects of the IFC and presentation techniques in the PTC. As discussed in (20), a conceptual object may be rendered with multiple presentation techniques. Therefore, consistency must be maintained between the multiple views of the conceptual object. Such mapping is yet another task of the dialogue controller.

The Dialogue Controller at Multiple Resolution

Bridging the gap between the IFC and the PTC covers task sequencing, formalism translation, and data mapping. Experience shows that these operations must be performed at multiple levels of abstraction and distributed among multiple agents.

Levels of abstraction reflect the successive operations of abstracting and concretizing. Abstracting combines and transforms events coming from the presentation techniques into higher level events for higher abstractions. Conversely, concretizing decomposes and transforms high level information into low level information. The lowest level of the dialogue controller is in contact with the presentation techniques provided by the toolkit. This boundary is a rather fuzzy frontier.

In general, user interface toolkits such as the X Intrinsic, provide an abstraction mechanism for defining new interaction techniques. However, it is not always possible to build new interaction techniques from the predefined building blocks of the toolkit. For example, in an earlier version of the X intrinsic, interaction techniques would occupy rectangular areas only. In such conditions, the notion of a wall in a floor plan drawing editor, could not be implemented as a diagonal line widget. Instead, a presentation object "wall" would be defined as a new abstraction in the dialogue controller portion.

This example shows that the frontier between the dialogue controller and the presentation techniques component depends on the facilities provided by the user interface toolkit and the presentation requirements of the interactive system: specific presentation requirements that can be built from the building blocks of the toolkit belong to the interaction techniques component. Those which cannot be built with the toolkit are part of the dialogue controller.

As discussed above, the multi-agent approach is a promising way to support parallelism, distribution, multi-thread dialogues, and iterative design. Since agents should carry task sequencing, formalism transformation, and data mapping at multiple levels of abstraction, it is tempting to describe the dialogue controller at multiple grains of resolution combined with multiple facets.

At one level of resolution, the dialogue controller appears as a "fuzzy potato". At the next level of description, the main agents of the interaction can be identified. In turn, these agents are recursively refined into simpler agents. This description is nothing more than the usual abstraction/refinement paradigm applied in software engineering. Orthogonal to the refinement/abstraction axis, we introduce the "facet" axis. As described in the next section, an agent is also described along three facets: Abstraction, Presentation, Control. These facets are used to express different but complementary and strongly coupled computational perspectives.

Figure 3 shows the recursive description of the dialogue controller at multiple grains of resolution combined with the 3 facets dimension.

4. A PAC Agent

In general, a PAC agent in the dialogue controller is composed of three facets (for a more complete description of the PAC model one can refer to (22) chapter 8, pp 161):

- The Presentation implements the perceivable behaviour of the agent. It is related to some presentation technique(s) of the PTC.
- The Abstraction implements the competence of the agent (i.e. its expertise) in an essentially media independent way. It maintains the abstract state of the agent and defines an interface with other agents. It may be related to some conceptual object(s) of the IFC. The abstraction facet defines the location for performing semantic delegation.
- The Control part of an agent is in charge of two functions: linkage of the Abstraction part of the agent to its Presentation portion, and maintenance of the relationships of the agent with other agents. The linkage serves two purposes: formalism transformations between the Abstraction and the Presentation portions of the agent, and data mapping between the abstract facet and the presentation facet. Relationships between agents may be static or dynamic. Dynamic relationships are required when

agents are dynamically created/deleted. Relationship maintenance by the control part of an agent covers the communication and the synchronization mechanism between this agent and its cooperating partners.

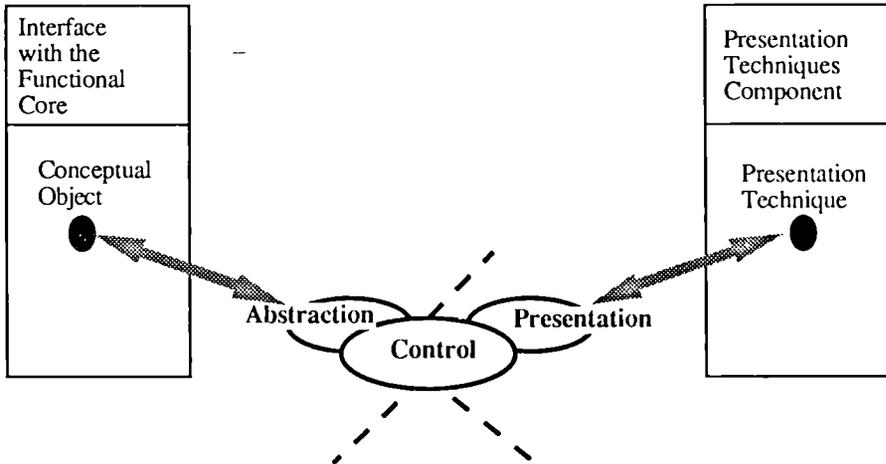


Fig. 4: A PAC agent of the Dialogue Controller. Dashed lines represent possible relationships with other agents. Dimmed arrows show the possible links with the surrounding components of the dialogue controller.

Figure 4 shows how a PAC agent relates to other PAC agents and to the surrounding world of the dialogue controller: conceptual objects in the IFC and presentation techniques in the PTC. If we consider the dialogue controller as a whole:

- the set of Abstraction parts of the various agents defines the internal state of the interaction,
- the set of Presentation parts defines the external state of the interaction, and
- the set of Control parts defines the mapping functions between the internal and the external state. Some properties of these functions are defined in (14).

This general description of the dialogue controller does not provide enough insight about how to define the agents for a particular interactive system. This is the topic of the next section.

5. Inside the Dialogue Controller: from Model to Reality

The picture shown in Figure 4 is an idealized view of the reality. Experience shows that some agents do not require all the facets advocated by the model. Early experiences with PAC has lead us to define a taxonomy for PAC agents as well as guidance for the abstraction/refinement process of agents. The result of our reflection is presented in the following two paragraphs.

A dimension space for PAC agents

PAC agents may be classified along two dimensions: linkage with conceptual objects and facets.

Linkage with conceptual objects

An agent may or may not be linked to a conceptual object. When linked to a conceptual object, it participates in a chain of data transformations and data mappings. This chain ends with the agent whose presentation part delivers the conceptual object to the user.

Agents that are not linked to any conceptual object are not domain dependent but they support the accomplishment of the task. They are a form of semantic enhancement delegated into the user interface. For this reason, we call them "syntactic agents". For example, in a visual programming environment system, the functional core maintains the abstract tree of the program being built whereas an agent in the dialogue controller may be used to trace program execution.

Facets of a PAC agent

A PAC agent always includes a control part. Since it links an agent to its environment, a control part necessarily plays a key role in the existence of an agent. PAC agents, however, need not include either an Abstraction facet or a Presentation facet.

Empty Abstraction occurs in two circumstances:

- 1) the agent, which is used as a presentation object only, has no particular competence. It may result from an extension of the toolkit unable to provide the appropriate facilities for defining a new interaction technique;
- 2) the agent is in direct correspondence with a conceptual object. In order to avoid duplication of information, the abstract competence of the agent is left in the conceptual object. In such a case, the abstraction is limited to the identification of the conceptual object.

Agents may have no presentation. As in the case of *view controllers* in Serpent (23), they are used as computational units to maintain relationships between multiple agents. In the next section, we discuss a number of such relationships.

Rules for identifying PAC agents

Agents may be used to represent combinations of relationships and levels of abstraction. We provide a set of rules that may help in the design process of building dialogue controllers in terms of agents. A more complete description can be found in (24).

Distributed Syntax

Use an agent to cement actions distributed over multiple agents into a higher abstraction.

It is often the case that the specification of a command involves actions distributed over multiple agents. This is particularly true for CAD application systems where a palette, which is an agent, displays the possible options and where a window, which is yet another agent, is used as a scratch area to create and edit an artefact (e.g. a picture in a graphic editor). For example, drawing a circle in an interactive system such as MacDraw implies selecting the circle icon in the palette then drawing the circle in the scratch area. The combination of the two actions has a meaning: that of creating a circle but the palette and the scratch area are not aware of each other. They would otherwise be dependent on each other and thus the first one would not be reusable without the existence of the second one. In addition, multi-thread dialogue authorizes logically connected actions to be interleaved with "foreign" actions. As a consequence, there is a conflicting need for maintaining "mutual ignorance" between two agents, for remembering logically connected actions and for combining them into a higher abstraction (e.g. a command). The multi-agent framework provides a natural way for satisfying such requirements: introducing an intermediary agent which serves as a cement. Figure 5 shows the general solution.

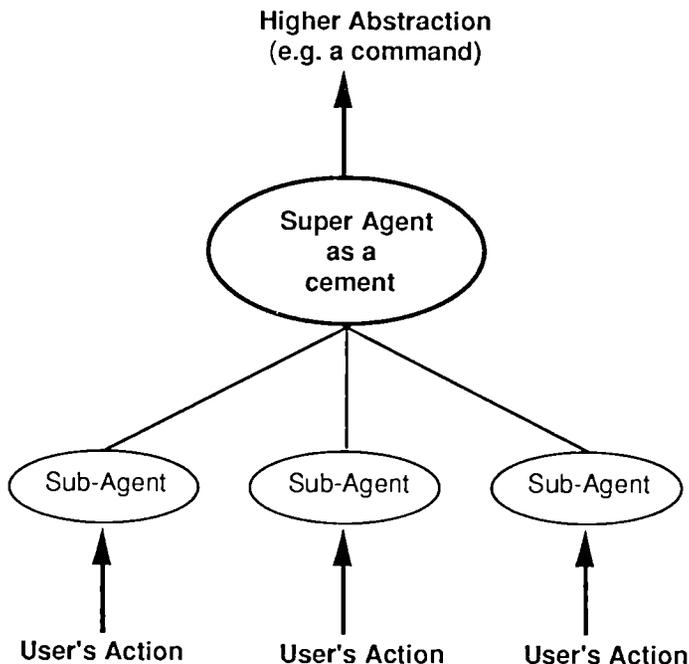


Fig. 5: Use of an agent to cement logically connected and distributed actions into a higher abstraction.

The cement agent behaves like a syntax analyzer which controls the local automata maintained by its subagents. By doing so, input for syntax analysis may come in any order from multiple sources. If, in addition, the cement agent is able to handle the

construction of several commands simultaneously, it provides an easy way to implement multi-thread dialogues.

Multiple Views

Use an agent to maintain visual consistency between multiple views.

One semantic concept of general interest is that of "focus of attention". In a document preparation system, the "focus of attention" is a document. The problem with the existence of multiple views on the same concept is the maintenance of visual consistency. Any action on one view should be reported in the other views.

According to the software principle which stresses "mutual ignorance" to enhance reusability, agents which implement the views of a semantic concept should not know each other. As shown in Figure 6, a "Multiple view" agent is introduced to express the logical link. Any action with visual side effect on a view is reported to the Multiple view which broadcasts the update to the other siblings.

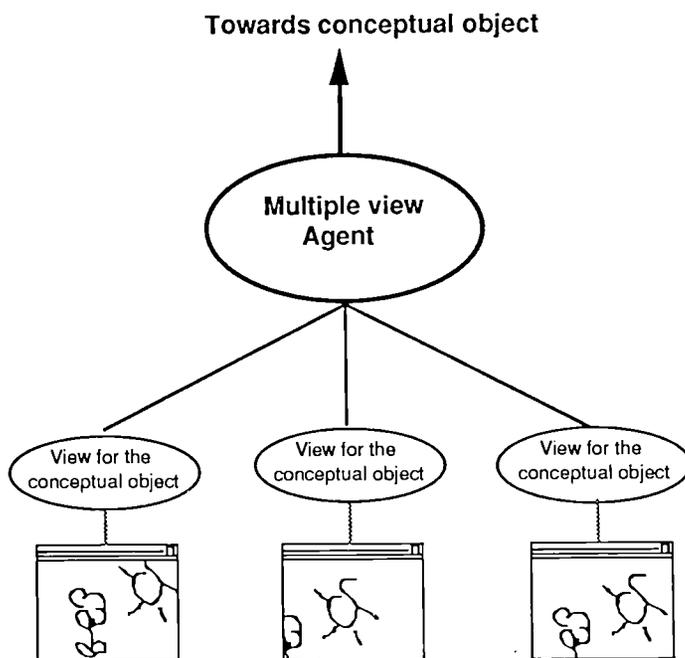


Fig. 6: Use of an agent to maintain visual consistency between multiple views of the same conceptual object.

Conveying composed conceptual objects

Use isomorphic agents to reflect the composition of conceptual objects.

It is often the case that conceptual objects are composed of conceptual objects. For example, a program is composed of modules which in turn is composed of procedures, etc. If the relation of composition must be conveyed to the user, then one can define a similar composition in terms of agents in the dialogue controller. A hierarchy of agents isomorphic to the hierarchy of conceptual objects is thus defined.

6. Conclusion

We have presented some preliminary results related to our heuristic approach to the design of interactive software architectures. Our contribution is three-fold:

- 1) We have clarified the description of the overall architecture of an interactive system making explicit the boundaries with the functional core and the presentation toolkit, two components considered by programmers as unavoidable constraints.
- 2) We have refined the dialogue controller into a number of agents modelled along two dimensions: facets and levels of abstractions. The combination of these two axes is a convenient way of describing phenomena observed from current practices.
- 3) We have identified a number of heuristic rules which provide useful insights and guidance for the definition of relationships between agents.

In the near future, we will formalize our design heuristics rules and develop an expert system which will be able to automatically suggest the agents appropriate for a particular system.

Acknowledgments

This paper was influenced by stimulating discussions with our colleagues and observers of the IFIP WG 2.7, with members of the AMODEUS project: G. Abowd, Bala, L. Bass, M. Beaudouin-Lafon, M. Harrison, I. Newman, P. Dewan, J. Larson, C. Unger, H. Stiegler. A. Chabert of the IHM group of LGI and V. Normand of Bull-Imag also deserve special thanks for their contribution to our model.

References

- (1) OSF/Motif, Programmer's Reference Manual, Revision 1.0; Open Software Foundation, Eleven Cambridge Center, Cambridge, MA 02142, 1989.
- (2) K. Schmucker. MacApp: An Application Framework; Byte 11(8), 1986, pp. 189-193.
- (3) B. F. Webster. The NeXT Book; Addison Wesley, 1989.
- (4) R.W. Scheifler, J. Gettys. The X Window System; ACM Transactions on Graphics 5(2), April, 1986, pp. 79-109.
- (5) User Interface Management Systems; G. E. Pfaff ed., Eurographics Seminars, Springer-Verlag, 1985.
- (6) D.A. Duce, M.R. Gomes. Hopgood, Lee (eds): User Interface Management and Design; Proceedings of the Workshop on UIMS and Environments, Lisbon, June 1990; Springer Verlag Publ., 1991.
- (7) L. Bass, R. Little, R. Pellegrino, S. Reed, R. Seacord, S. Sheppard and M. R. Szezur. The Arch Model: Seeheim Revisited; User Interface Developers' Workshop; April 26, 1991, Version 1.0.
- (8) J. Coutaz. UIMS, Promises, Failures and Trends; HCI'89, People and Computers V, A. Sutcliffe, L. Macauley eds., Cambridge University Press, 1989, pp. 71-84.
- (9) J. Coutaz. PAC, an Implementation Model for Dialog Design; Interact'87, Stuttgart, September, 1987, pp. 431-436.
- (10) A. Goldberg. Smalltalk-80: The Interactive Programming Environment; Addison-Wesley Publ., 1984.

- (11) ESPRIT BRA 3066. Assimilating Models of Designers, Users and Systems (AMODEUS), Technical Annex, CEC, Brussels, 1989.
- (12) R. Young, P. Barnard. The Use of Scenarios in Human Computer Interaction research: Turbocharging the tortoise of cumulative science; In Proceedings of the CHI+GI'87 Conference (Toronto); ACM, 1987, pp. 291-296.
- (13) N. Hammond, P. Barnard, J. Coutaz, M. Harrison, A. MacLean, R. Young. Modelling User, System and Design: Results of a Scenarios Matrix Exercise; Panel in Proceedings of the CHI'91 Conference (New Orleans, 28 April-2 May); ACM, 1991, pp. 377-380.
- (14) M. Harrison, H. Thimbleby (eds). Formal Methods in Human Computer Interaction; Cambridge University Press, 1990.
- (15) P. Barnard. Cognitive Resources and the Learning of Human Computer Dialogs; in *Interfacing Thought, Cognitive Aspects of Human Computer Interaction*, J.M. Carroll Ed., MIT Press Publ., 1987, pp. 112-158.
- (16) R.M. Young, T. Green, T. Simon. Programmable user models for predictive evaluation of interface design; In Proceedings of the CHI'89 conference, 1989, ACM press, pp. 15-19.
- (17) M. Harrison, G. Abowd. Formal methods in Human Computer Interaction; Tutorial #27, Computer Human Interaction Conference, 1991.
- (18) D. A. Norman, S. W. Draper. *User Centered System Design*; Lawrence Erlbaum Associates, Publishers, 1986.
- (19) P. Tanner, W. Buxton. Some Issues in Future User Interface Management Systems (UIMS) Development. IFIP Working Group 5.2 Workshop on User Interface Management, Seeheim, West Germany, November, 1983.
- (20) Coutaz, S. Balbo. Applications: a dimension space for user interface management systems; In Proceedings of the CHI'91 Conference (New Orleans, 28 April-2 May); ACM, 1991, pp. 27-32.
- (21) J. Bass, J. Coutaz: *Developping Software for the user interface*; Addison Wesley, 1991.
- (22) J. Coutaz. *Interfaces Homme-Ordinateur: Conception et Réalisation*; Dunod, 1990.
- (23) L.J. Bass, E. Hardy, K. Hoyt, R. Little, R. Seacord. The Serpent run time architecture and dialogue model; Carnegie Mellon Technical Report, CMU/SEI-88-TR-6, January, 1988.
- (24) *Software Architecture Modelling*, RP2, Deliverable D2, september, 1990.

DESIGN SPACE ANALYSIS: BRIDGING FROM THEORY TO PRACTICE VIA DESIGN RATIONALE

ALLAN MACLEAN, RICHARD YOUNG¹,
VICTORIA BELLOTTI and THOMAS MORAN²
Rank Xerox EuroPARC
61 Regent Street
Cambridge CB2 1AB
England

¹ Also at MRC Applied Psychology Unit,
15 Chaucer Road, Cambridge, England

² Now at Xerox Palo Alto Research Center,
3333 Coyote Hill Road, Palo Alto CA 94304, U.S.A.

Email: maclea.europarc@rx.xerox.com

SUMMARY

This paper reports work being carried out under the AMODEUS project (BRA 3066). The goal of the project is to develop interdisciplinary approaches to studying human-computer interaction and to move towards applying the results to the practicalities of design. This paper describes one of the approaches the project is taking to represent design - Design Space Analysis. One of its goals is help us bridge from relatively theoretical concerns to the practicalities of design. Design Space Analysis is a central component of a framework for representing the design rationale for designed artifacts. Our current work focusses more specifically on the design of user interfaces. A Design Space Analysis is represented using the QOC notation, which consists of *Questions* identifying key design issues, *Options* providing possible answers to the Questions, and *Criteria* for assessing and comparing the Options. In this paper we give an overview of our approach, some examples of the research issues we are currently tackling and an illustration of its role in helping to integrate the work of some of our project partners with design considerations.

1. DESIGN SPACE ANALYSIS

AMODEUS (Assimilating Models of Design, Users and Systems) is a project which brings together teams from different disciplines with three general objectives (1) to extend the scope of modelling techniques to provide analytic leverage on the problems of user-system interaction: (2) to bridge the conceptual gaps between behavioural and computing disciplines: and (3) to bridge from theory to the practicalities of designing software artifacts. This paper reports work aimed at supporting the third of these objectives. The work reported has two goals. The first is to develop a technique for representing design decisions which will, even on its own, support and augment design practice. The second goal is to use the framework as a vehicle for communicating and contextualising more analytic approaches to user-system interaction into the practicalities of design. However, we believe that the second of these goals cannot be

achieved without also achieving the first. This paper describes some aspects of Design Space Analysis, the basis of our framework for representing design, some prospects for it being the basis of tools for supporting design in the future, and its relationship to the other activities taking place within the AMODEUS project.

Design Space Analysis is an approach to representing design rationale (10, 12). It is a central part of a long term project in which we are interested in helping software designers reason about design (individually and in groups) and produce an output which can help others to understand why the resulting design is the way it is. Although our intent is to develop a framework for design applicable to a variety of domains, much of our work to date has been aimed at HCI audiences for two main reasons. First, the design domain in which we are most actively working is user interface design. Secondly, we believe that HCI expertise is a crucial ingredient in developing the approach itself as the usability of the tools and techniques we produce will ultimately be critical in determining the acceptability of our work to its users - i.e. the designers of computer systems.

A key characteristic of our approach is that the output of design is conceived of as a *design space* rather than a single artifact. The approach therefore contrasts with the traditional conception of design which assumes that the eventual output is a specification or artifact. The final artifact, although embodying the designer's decisions, does not preserve any of the thinking and reasoning which went into its creation. We use a semi-formal notation (called *QOC*, for Questions, Options & Criteria) to represent the design space around an artifact being produced. This design space is an explicit representation of alternative design options, and an explicit representation of reasons for choosing among those options. The main concepts we use for the representation are *Questions* which highlight key issues in the design, *Options* which are effectively answers to the Questions and *Criteria* which are the reasons that argue for or against the possible Options. Figure 1 illustrates the relationship between these constituents of the design space.

This representation of a design space provides a succinct rationale for the final design by placing it in a broader context which highlights how it might be different and why it is the way it is. Such a representation should be able to support communication between people with different backgrounds and goals, for example between members of a design team working on an initial design, between the original designers and designers of a later generation system who want to re-use parts of the original design, and even between the designers and users of a computer system. Exploring these claims is a future part of our research strategy. However, Design Space Analysis provides a theoretical framework for design which we have already found useful for helping us better understand design issues (10, 11, 12, 13). This theoretical approach drives such activities as understanding the design process and how it can be improved (11); requirements for tools to support the creation of a QOC representation (and thus the design process) (12); and the integration of other approaches, such as cognitive and system HCI modelling techniques, into software design (1). We believe that our approach has the potential of supporting designers with techniques and ways of working which will be useful even in the absence of computer based tools (11, 12). However, if appropriate tools can be devised more powerful support should clearly be possible.

In this paper we briefly discuss three of these perspectives. First we illustrate design reasoning. We then highlight some of the most difficult issues which our experience so far leads us to believe need to be tackled to provide adequate computer based support

for Design Space Analysis. Finally we illustrate the relationship between Design Space Analysis and HCI modelling techniques within the context of the AMODEUS project.

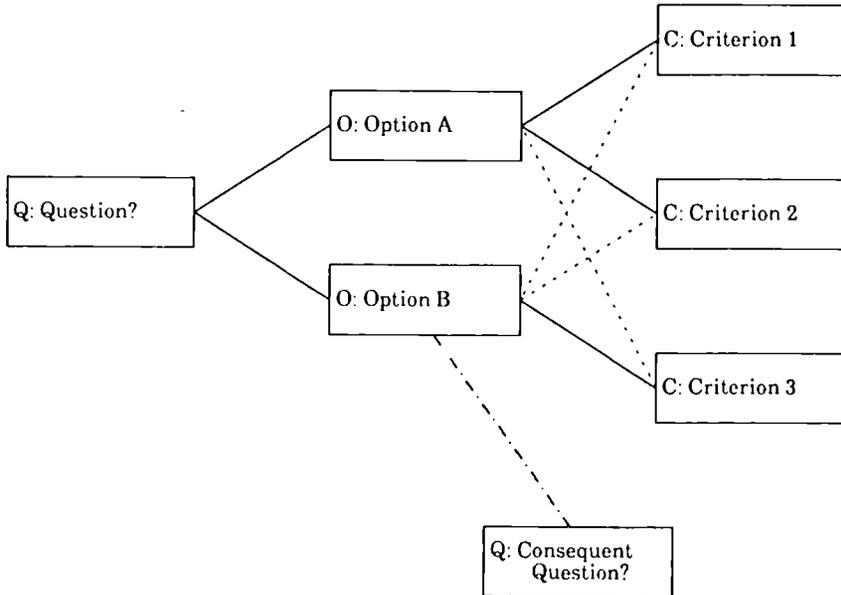


Figure 1. The components of a design space. Options can be thought of as "answers" to Questions. Questions highlight important dimensions in the design space. Criteria argue for or against possible Options. We find that considerable power can be gained from regarding relationships between Options and Criteria as relatively positive (solid line between Option and Criterion) or negative (dotted line) - i.e. arguing for or against the Option. Options may spawn off consequent Questions which allow more detailed aspects of the design to be addressed.

2. REASONING IN A DESIGN SPACE

In this section we illustrate design reasoning within part of a design space represented using the QOC notation. The example is based on a small issue in the design of a bank ATM (Automated Teller Machine) user interface. The course of reasoning is hypothetical, but the content is based on some of our own analyses of the ATM design space, and on insights from designers we have observed working on the same problem (11).

For present purposes, let us consider only one very small part of the ATM design space - when to return the ATM card to a customer. The following example illustrates this Question along with a matrix of possible Options (i.e. possible "answers" to the question), some Criteria which are relevant for evaluating these Options, and an indication of whether each Option is good (+) or bad (\$) relative to each Criterion. The precision with which Criteria are related to Options could easily be greater - e.g. by using a five point scale. However, for an initial qualitative exploration of the design space, a simple binary distinction as we use here is often sufficient. For more general purposes, and for larger design spaces, we would normally represent the design space as set of nodes and links, perhaps using a hypermedia system such as NoteCards (5)

- however, the visualisation shown in Figure 2 gets the main points of present interest across, and can in fact be useful for reasoning about local parts of a design space.

CRITERIA			
OPTIONS	Bracket	Speed	Security
Immediately after inserting card	-	+	-
At end of all transactions	+	-	+

Figure 2. A piece of design space to help reasoning about the Question "When should the ATM card be returned to the customer?"

The two Options being compared initially in Figure 2 are: to return the card to the customer immediately after inserting it, or to return it as the final part of the transaction. The *bracket* Criterion claims that since the user initiates the interaction by inserting the card, removing the card is an appropriate action for terminating the interaction as the card can then be perceived as a token bracketing the interaction with the machine. The *speed* Criterion claims that if the user deals with the card only once (inserting it and removing it immediately afterwards), it will reduce the number of steps required to use the machine because inserting and removing the card becomes one step, and so immediate return is the better Option. The *security* Criterion claims that returning the card at the end is preferable, on the grounds that the bank can retain the card if necessary, and information about the current transaction can be recorded on the card.

We would not of course necessarily want to conclude that the card should be returned at the end of the transaction since that option "wins" by two pluses to one. For example, we might want to stress the *speed* Criterion very strongly, and so go for the faster Option.

We may want to question the appropriateness of one of the Criteria being used. For example, for the *bracket* Criterion we may claim that the card is not the natural focus for the user of an ATM; people normally use such machines to get money; once they get the money they have achieved their goal in using the machine and are likely to walk away forgetting to remove their card. As a result of this argument, we may want to introduce a new Criterion which reflects compatibility with the users' perceptions of the task they are carrying out and which suggests that getting cash, not using an ATM card, is the task focus. Moreover, it would suggest that perhaps the evaluation on the *security* Criterion should be re-thought as someone else may pick up a card which has been left behind in error. Gaining a richer understanding of the relevant Criteria does not simply mean being better informed about which of the current Options to choose - it may help to generate alternative Options. For example, in this case we could add an Option to retain the card until all transactions have been specified, but to give back the card immediately before giving out the money, or perhaps to give back the card and money together from the same slot in the machine.

Note that the very nature of design (certainly the kind of design which we are considering) is that there is no one "best" solution. Using the QOC notation to represent

the design illustrates why this is so. The designer has two related tasks. One is to understand the relevant design space (potential options and reasons for choosing among them), and the other is to find an appropriate solution within that space. Since the space of possible options is effectively open ended, and is subject to constant change (e.g. new technology coming along, new ideas on the part of the designers, changing requirements from the users), the potential design space is in a constant state of flux and so a previously good solution within that space may become inappropriate. A discussion of some of the ways in which the representation of the space can be refined and reformulated is given in (3).

3. TOOLS FOR SUPPORTING DESIGN SPACE ANALYSIS

Our emphasis is to understand where current design practice can be improved, and where its characteristics place particular challenges for support. Part of our research strategy has been to produce Design Space Analyses ourselves. We have made use of simple document management tools such as text editors and drawing packages together with ubiquitous aids such as pencil and paper. The most sophisticated system we have employed has been NoteCards (5), used for holding representations of design spaces. A full analysis capturing a range of different kinds of information we have alluded to can become a large and messy object to draw on paper, so that appropriate computer support is essential if Design Space Analysis is to prove practicable for design practice. This Section discusses some of the challenges for such tools to be sufficiently usable and useful.

One problem with current tools which claim to help users structure ideas is that they force the user into making *premature commitments*. They insist on content being mapped into structure from the very beginning, before the user understands how best to characterise the ideas in terms of the structure. And once the structure is built, significant redundant effort is needed to change it (4, 14). This issue of premature commitment is widely recognised as a problem within, for example, the hypertext community (6). In the context of design, it causes difficulties. Even in our own analyses, it is not uncommon for our initial conceptualisation of part of the design to prove inappropriate. Thus, in a tool for supporting the manipulation of a QOC representation, it is highly desirable that the designer be allowed gradually to impose structure on initially unstructured material, and later to change it easily.

A second requirement of a tool for supporting Design Space Analysis is that it allow multiple views of the representation. In order to use it flexibly for different activities such as building new structure, browsing existing structure to understand the design, finding relevant Criteria, and so on, designers need to see views of the design space which are appropriate for the different tasks. They need to be able to move around, selecting different views as their understanding of the design changes. They may, for instance, want to filter the level of detail shown by specifying which types of objects and links should be displayed. Or it may be that a node-and-link style of presentation is not the most useful for certain purposes. For understanding the local evaluation of design Options, for example, they may want to select a particular Question and be shown an evaluation matrix of Options against Criteria, with each cell containing an indication of the link between them as we illustrated in the previous section. Alternatively, they may require a global view to see how one particular part of the design fits into the larger context.

If a tool had knowledge just of the QOC structure (as opposed to content), it could be able to offer low-level assistance to the designer in building the rationale. In (12), for

example, we offer a number of heuristics for constructing a design space and improving a design, some of which suggest that each Option should be connected to at least two Criteria, with at least one of the links being positive and one negative. A modestly "unintelligent" tool would be able to check that this heuristic had been followed, and otherwise suggest it as a piece of advice. This role of "helpful assistant" is analogous to that proposed by Young & Harris (15), who describe a structure editor for creating viewdata screens which helps to ensure that the range of necessary tasks are all undertaken. Similarly, Harp & Neches (8) point out that simply being able to detect the presence or absence of information can be used as the basis for simple automated reasoning, even if the information itself remains informal and its content therefore inaccessible to the machine.

4. DESIGN SPACE ANALYSIS: FROM MODELLING TO DESIGN

Figure 3 shows a characterisation of the AMODEUS project structure which emphasises the role the Design Space Analysis framework for design rationale plays in the project. By having a number of very different perspectives in one project each approach expects to benefit from the strengths of the others, and indeed the project can be viewed in ways which emphasise other relationships among the contributing partners. A brief description of this project will help to highlight how some of the challenges outlined above may be addressed.

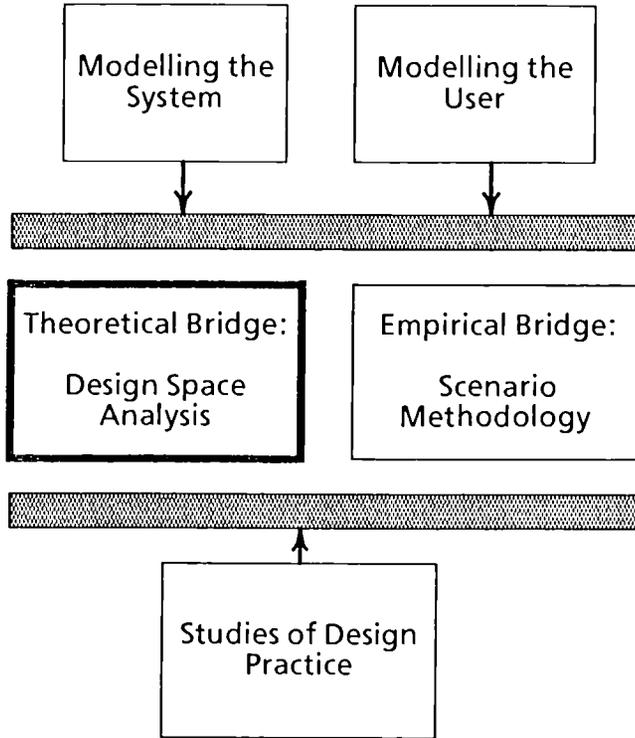
The project can be characterised in three distinct "layers" covering a spectrum from various approaches to modelling human-computer interaction which are relatively formal and limited in scope, through to studies of design practice where a wide range of relatively informal unstructured activities take place. The system modelling component is centred around work on formal specifications from the computer science tradition (e.g. 9). The user modelling component comes from the psychological tradition and two approaches are represented in the project. One of these - PUMs (Programmable User Models) - is based around an AI problem-solving architecture (17). It is basically a problem-space model of the human cognitive architecture (based on SOAR) and embodies constraints on the representation, processing and acquisition of knowledge. The other is based on a cognitive resources approach, embodied within an expert system (2). The design study activity is geared towards exploring design practice with the aim of eventually incorporating the modelling activities into an applied setting (cf. 7).

Mediating between theory and practice, the scenario methodology (16) is aimed at developing a common set of exemplars for the different project activities to focus on, to improve communication between the various approaches. Finally, Design Space Analysis serves a role of providing a representation which helps mediate between the limited scope of the detailed modelling approaches and the relatively ill-structured breadth of design practice.

The AMODEUS project pushes our Design Space Analysis work in two different directions. First of all, the design study part of the project helps us compare our current conception of Design Space Analysis with design practice. For example the ATM design problem, on which the example described earlier was based, is helping us understand the extent to which examples of our concepts emerge naturally from designers working together. Understanding this relationship is important to ensure that human-human communication mediated by a design space representation will be fruitful. A corollary of this research strategy is that we do not believe that an approach which places excessive emphasis on preserving the richness of the design process will produce a

useful representation for design - the resulting representation would be excessively complex and unwieldy.

Theoretical Modelling



System Design

Figure 3. Schematic of the AMODEUS project, emphasising the influences on the Design Space Analysis component.

Secondly, the relationship with the modelling approaches has a role of developing Design Space Analysis in rather different directions. Our current conception emphasises that the design space itself has to be designed - it provides no assistance in helping constrain possible design options or relevant reasons for choosing among them. We are currently exploring the incorporation of aids to design into Design Space Analysis. Modelling approaches such as the ones being examined in this project are one important direction to pursue. The different modelling techniques incorporate a variety of approaches which assist with reasoning about design. By incorporating appropriate aspects of the modelling approaches into Design Space Analysis, we expect to assist the designer with the design task in the areas where the modelling techniques apply, while also providing an explicit context of the parts of the design where the modelling techniques do not apply.

5. AN EXAMPLE: THE UNDO SCENARIO

As a simple example of a relationship between the theoretical modelling approaches and Design Space Analysis, we will finish by briefly considering the analysis of a scenario which has recently been carried out within the AMODEUS project.

The scenario concerns the design of an undo facility in a multi-user text editor. The original scenario description is about one page of text, but the key points are as follows:

A team of designers is designing a multi-user text editor. They are considering two versions of how the undo facility should work:

- 1) Relative to the individual: each Users can UNDO only their own actions.
- 2) Relative to the document: actions can be UNDOne regardless of who carried out the original action.

The designers are also considering two versions of the software:

- A) The first release will have a *single* insertion point, with control moving between the users by explicit token passing.
- B) The second release will have *multiple* insertion points, so that each user can make changes simultaneously.

What can we advise them about the usability of the different designs?

A typical Design Space Analysis might start by simply structuring the basic information given in the design problem as shown by the Questions and Options in Figure 4.

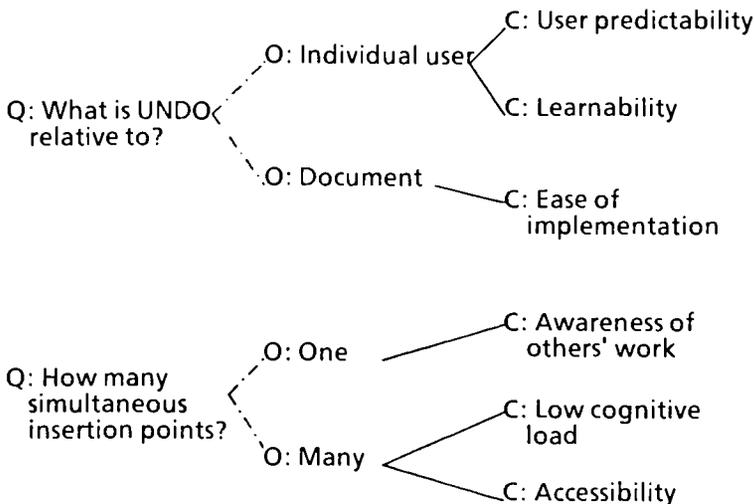


Figure 4. A Design Space Analysis of the UNDO scenario. Questions and Options are structured directly from the information in the scenario description. Criteria are derived from the analyses carried out by the theoretical modelling. Solid lines linking Options to Criteria represent positive assessments - i.e a claim that the Criterion supports the Option. (Negative assessments are not shown to minimise clutter on the Figure.)

The Criteria in Figure 4 represent some considerations from the analyses carried out by the theoretical modelling groups. The solid lines show which Options the Criteria are claimed to support. For example, the *user predictability* Criterion is derived from principles which the system modelling group can represent in a formal specification of the system behaviour. It basically claims that the user should be able to predict the effect of an undo action and that a system which carried out UNDO relative to the document as whole rather than the actions of an individual user would not meet the Criterion. The modelling argument is of course much more detailed than it is possible to do justice to here, but it should suffice to add that the further justification rests on considering what aspects of the document would be visible to the user and nature of transitions between system displays resulting from user actions.

Similarly, the *learnability* Criterion is based on the analyses from one of the user modelling groups. It also argues for UNDO being focussed around the actions of an individual user rather than the state of the document, but in this case the argument rests on the claim that learning how UNDO operates relies on immediacy in spotting the relationship between an action and an undesired result, and that this is not likely to be perceived unless the user was himself the originator of the action. These examples illustrate one role of the Design Space Analysis as showing how the different, more rigorous, analyses from the theoretical modelling groups relate to each other by mediating their claims via a representation of the design space. Further discussion of the justification of Option-Criterion links based on theory can be found in (12).

The various modelling approaches can clearly play an important role in helping to justify and evaluate candidate design Options. However, an alternative Design Space Analysis of the same scenario can be used to illustrate a more powerful role of user modelling in helping to drive design more directly. Rather than use the information contained in the design problem to impose initial structure on the design space, the user modelling analysis acts as the starting point. The PUMs knowledge analysis of the scenario identified four key pieces of knowledge which the user would need to operate an UNDO facility:

1. Know what the relevant stream of activity is.
2. Know how the stream is articulated into units.
3. Know which unit of activity is affected by UNDO
4. Know what the effect of UNDO is on that unit.

We have used these key knowledge requirements to suggest QOC Questions to "seed" the design space. Figure 5 shows part of a Design Space Analysis based on these Questions. The aim here is simply to illustrate the rather different emphasis which this orientation gives the design space, so Criteria have not been represented and we do not intend to provide a detailed discussion of the Options represented. A point to note is that the Questions are much more user-oriented than those used in Figure 4. This is particularly important in helping maintain a user-oriented view from the outset by structuring the view of the design space around considerations relevant for the user. In addition, since the Questions are not simply structuring information already given, they serve the role of opening up the design space by helping to suggest a much broader range of possible design solutions, and help get at more detailed aspects of the design which may be necessary to gain a better understanding of the key issues.

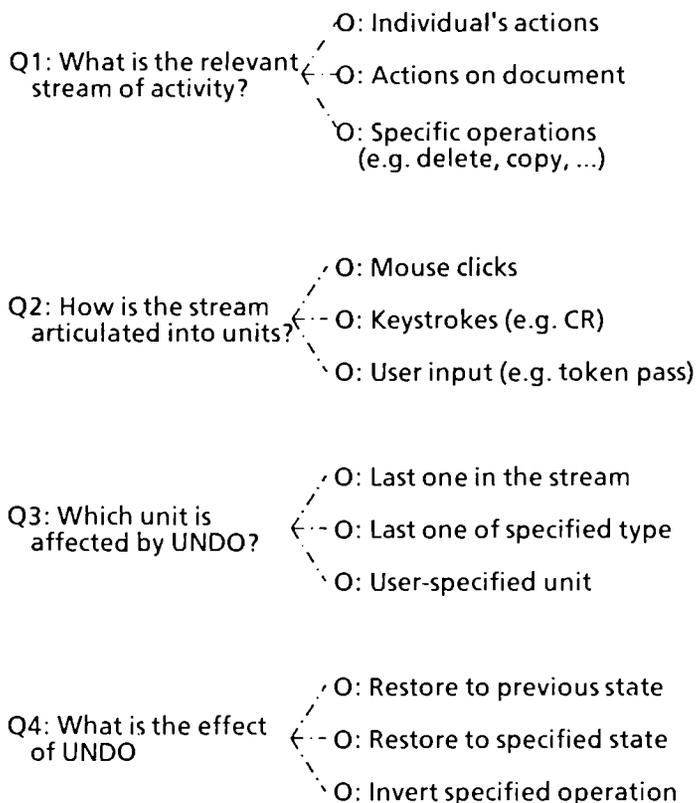


Figure 5. Part of a Design Space Analysis which starts from Questions generated by the PUMs knowledge analysis. The Options are design possibilities suggested by the Questions. (Criteria to help evaluate the Options are not represented.)

6. CONCLUSIONS

We have presented an overview of the Design Space Analysis component of the AMODEUS project and illustrated some of the problems and prospects for developing it into a technique for supporting software design. More specifically, we have also illustrated how the approach relates to the range of activities being explored in the AMODEUS project and some of the mutual benefits which results from these activities being carried out within a single project. Theoretical modelling activities provide more rigorous analysis than is possible with Design Space Analysis on its own, and Design Space Analysis provides a mechanism for grounding these analyses within the details of design.

REFERENCES

- (1) AMODEUS (1989). Applying Models Of DEsign, Users and Systems (AMODEUS). Technical Annexe. Esprit Basic Research Action 3066, Brussels.
- (2) BARNARD P., WILSON, M. and MACLEAN, A. (1988). Approximate modelling of cognitive activity with an expert system: A theory based strategy for developing an interactive design tool. *The Computer Journal*, 31, 445-456.

- (3) BELLOTTI, V., DOURISH, P. and MACLEAN, A. From Users' Themes to Designers' DRreams: Developing a Design Space for Shared Interactive Technologies. AMODEUS RP6/WP7.
- (4) GREEN, T. R. G. (1989). Cognitive dimensions of notations. In A. Sutcliffe & L. Macaulay (Eds.), *People and Computers V: Designing for Usability*, Proceedings of HCI'89, September, Nottingham, 443-460. Cambridge: Cambridge University Press.
- (5) HALASZ, F., MORAN, T. and TRIGG, R. (1987) NoteCards in a Nutshell. In Proceedings of CHI + GI'87: Human Factors in Computing Systems, Toronto, Canada. ACM, New York, 45-52.
- (6) HALASZ, F. (1988) Reflections on NoteCards: Seven Issues for the Next Generation of Hypermedia Systems. *Communications of the ACM*, 31, 836-852.
- (7) HAMMOND, N., JRGENSEN, A.H. , MACLEAN, A., BARNARD, P. and LONG, J. (1983) Design Practice and Interface Usability: Evidence from Interviews with Designers. In Proceedings of CHI'83: Human Factors in Computer Systems, Boston. ACM, New York, 40-44.
- (8) HARP, B., and NECHES, R. (1988). Notecards: An everyday tool for aiding in complex tasks. In *Collected Papers of the Workshop on Architectures for Intelligent Interfaces: Elements and Prototypes*, Asimolar, California, 287-304. ACM/SIGCHI.
- (9) HARRISON, M. ROAST, C. and WRIGHT, P. (1989). Complementary Methods for the Iterative Design of Interactive Systems. In Proceedings of HCI International, Boston.
- (10) MACLEAN, A., YOUNG, R., and MORAN, T. (1989). Design Rationale: The argument behind the artifact. In Proceedings of CHI'89: Human Factors in Computing Systems, April 30 - May 4, Austin, Texas, 247-252. New York: ACM.
- (11) MACLEAN, A., BELLOTTI, V., and YOUNG, R. (1990). What rationale is there in design? In Diaper, D., Gilmore, D., Cockton, G. and Shackel, B. (Eds.) *Proceedings of Interact'90 Third IFIP Conference on Human-Computer Interaction*. (Aug 27-31, 1990, Cambridge, England). Amsterdam, Elsevier North-Holland, pp207-212.
- (12) MACLEAN, A., YOUNG, R., BELLOTTI, V. and MORAN, T. (1991) Questions, Options and Criteria: Elements of Design Space Analysis. *Human Computer Interaction*, 6 (3 & 4).
- (13) MACLEAN, A., BELLOTTI, V., YOUNG, R. and MORAN, T. (1991) Reaching through analogy: A Design Rationale perspective on roles of analogy. In Proceedings of CHI'91: Human Factors in Computing Systems, April 28 - May 2, New Orleans, Louisiana. 167-172. New York: ACM.
- (14) SHUM, S. (1991) Cognitive Dimensions of Design Rationale. In *People and Computers VI* (eds.) D. Diaper and N. V. Hammond. Cambridge: Cambridge University Press.
- (15) YOUNG, R. M., and HARRIS, J. E. (1986). A viewdata-structure editor designed around a task/action mapping. In M. Harrison & A. Monk (Eds.), *People and Computers: Designing for Usability*, Proceedings of HCI'86, September, York, University of York, 435-446. Cambridge: Cambridge University Press.
- (16) YOUNG R. and BARNARD, P. (1987) The Use of Scenarios in Human-Computer Interaction Research: Turbocharging the Tortoise of Cumulative Science. In Proceedings of CHI + GI'87: Human Factors in Computing Systems, Toronto, Canada. ACM, New York, 21-26.
- (17) YOUNG R., GREEN, T. and SIMON, T. (1989). Programmable User Models for Predictive Evaluation of Interface Designs. In Proceedings of CHI'89: Human Factors in Computing Systems, Austin, Texas. ACM, New York, 15-20.

MAPS AS BULK TYPES FOR DATA BASE PROGRAMMING LANGUAGES

Malcolm Atkinson*
Christophe Lécluse† Paul Philbrow‡
and
Philippe Richard†

Abstract

A new bulk type constructor, **map**, for database programming languages is introduced. It is designed to balance the requirements for persistent canonical store implementation with the requirements of data modelling. A map value denotes a stored finite updateable function, and may be considered as a set of pairs of tuples, denoting the domain and range respectively. The map includes relations and sets as a degenerate form, when the range is empty, and the map algebra is then equivalent to relational algebra. Utilising an identified range allows the map to model directed structures such as graphs, which have been explored hitherto in the functional data models. Maps have a user-specifiable equality test on domain elements, and a user-specifiable stored ordering.

A map will trivially represent sets, arrays, sparse arrays, index structures, and relations; hence it is a candidate for a canonical store. Extended to polymorphic maps, it will also represent records, environments and strongly typed memo-functions and command-histories. Inter-operability between various languages is envisaged via such a canonical store. The map construct is sufficiently high-level and abstract that it will convey the structural semantics between the inter-working components. This same captured structure makes it possible to develop efficient map stores utilising indexing, clustering and prefetching. Both explicitly controlled operations, and an optimisable algebra are defined for maps.

Keywords: data modelling, database programming languages, bulk data, data types.

1 Introduction and Motivations

Database programming languages aim to reconcile database and programming language technologies. A crucial step toward this integration is to find constructs unifying data models, as used in information system modelling, and type systems, as used in programming languages.

In existing database programming languages like DBPL [MS89], Napier88 [MBC+88] [MBC+90], Galileo [ACO85] or O_2 [LR89], this integration is provided through a type construction defining instances that are arbitrarily large collections of elements. Such types, called *bulk types*, are of interest both in programming languages and in data models [ART91].

*University of Glasgow, Department of Computing Science, Glasgow G12 8QQ, Scotland. Parts of this work were done when M. Atkinson was visiting Altaïr.

†GIP Altaïr, INRIA, BP105, 78153 Le Chesnay, France.

‡University of Glasgow

It is possible in modern programming languages to define representations capable of holding a bulk type, for example lists, but these will often fail to capture important semantics - for example, that the list is a set. It is also difficult, without new notations, to provide elegant and efficient ways of describing operations involving these types or generating instances of these types. Notations using libraries that provide bulk types tend to be much more verbose and are less amenable to global optimisation. In general, they do not communicate information to the storage system that can be used for establishing consistent inter-operability, or for establishing efficient data storage. On the other hand, built-in constructs may be less flexible and more expensive to provide [MS91].

In database modelling, the primary interest is adequate description of the semantics of the type. Generally this is achieved by the database designer choosing and building into the data model a small number (often just one) of bulk data types. For those designers, new forms of more general modelling abstractions, may be of interest, particularly if, without adding complexity, they increase the power of the model to describe structure.

We observe that the normal treatment of bulk data types at present, both in data models and in programming languages, is for the designer to build in a few, and hope this satisfies the users. It rarely does. The built-in constructs are usually complex and difficult to understand. Their interaction is often harder to understand. Yet the provided set of bulk types may match poorly with the structures which people wish to model, store and manipulate. Even semantic data models [IK87, PM88] generally provide a restricted set of constructs for bulk data.

For example, suppose that the designer has provided sets and sequences, as in O_2 . The user then wants to have bags and directed acyclic graphs. This leads the designer to give up. Adding more things would make the language or data model unduly complex, and the consumer is unlikely to be satisfied. Applications using such components still get built because of the ingenuity and perseverance of the advanced programmers working on the project. But there are never enough of these “advanced programmers”.

An alternative, or complementary strategy, is to provide facilities in the language for bulk types to be defined. This allows the language to be kept simple, and “primitive” types to be combined to provide the bulk types. But this path also has its difficulties. The effort of designing and implementing good quality bulk types is high. It would be the task of those advanced programmers to define these types. To amortise their efforts effectively, we should put these types in a library. Then the question arises: can they be easily extracted and used by the average application programmer? We believe that the difficulty of understanding their semantics should not be significantly different when the types are in a library or when the same types are built-in. However the notations for using library types and instances of types may be much more cumbersome than built-in notations, which may also be suggestive of the type’s semantics. Furthermore, the parameterisation and polymorphism available over types in the library may be less than that for built-in types. For example, the construct presented in this paper would, in present languages, require an infinite set of library map constructors to match the model we offer.

We therefore present maps for three reasons:

1. As a challenge to language and data model designers, since maps typify the kind of bulk type which it is difficult to parameterise and use in existing data models and languages;
2. As a prototypical design of a powerful bulk type that might be provided either as a built-in type, or, if the language designers rise to the variability challenge, as a library type;
3. As a potential “type-quark” which will allow data models and programming languages to be simplified by unifying constructs; and,

4. As a canonical store construct suitable as one of the repertoire of constructs in a persistent object store serving a variety of languages. We think that this construct can be a step toward interoperability of heterogeneous database or persistent programming languages at the store level. We have already shown in a separate report [ALPR91b] how maps can simulate the constructs found in persistent languages such as Napier88 [MBC+90, DCBM89].

This paper is organised as follows: Section 2 introduces the map construct through examples and shows their expressive power. Section 3 is devoted to the presentation of the map algebra and shows that relational or object-oriented queries can be expressed using it. Section 4 illustrates the modelling power of maps. Section 5 makes a comparison between maps and related constructs in programming languages such as MUMPS and DBPL. Finally, we conclude and give directions of research that we are currently investigating. A full example of programming a bibliographic application is provided in the appendix.

2 Presentation of Maps

We do not present a new database programming language here. We present a type constructor, called `map`, which we claim is a good candidate as a bulk type for any database programming language. In the examples, we will use a syntax which is an extension of the database programming language Napier88, but several implementation strategies for maps are possible and under investigation.

Since bulk types are a major rôle for the map construct we first list, in Section 2.1, the requirements for bulk types. Some caveats and desiderata may be found with more details in [ART91].

2.1 Requirements for Bulk Types

The requirements are not necessarily independent of one another, nor are they presented in some order of priority, and they may conflict, requiring trade-offs during design, and variations in implementation. These criteria are not exclusive to bulk types.

1. Instances of a bulk type are potentially large;
2. Instances of a bulk type may vary in size (length or cardinality), i.e. different instances of the same type may have different size, and for some bulk types the size of a given instance may change as a result of updates, a consequence of this property is that the size of an instance is not normally reflected in its type definition;
3. Instances of a bulk type may be mutable, i.e. it should be possible to alter the value of a bulk type without altering its identity (references to the original value will now reference the amended value);
4. A useful and succinct algebra, with well understood properties, is required over the instances of the bulk types;
5. Data type completeness is expected, that is the elements of a bulk data type should be parameterisable with any type that has minimum required properties for the structure of that bulk data type;
6. Bulk data types should be first class citizens in the language, that is all the operations that are universally allowed on other types should also be available for these types, for

example, being assigned, being parameters and results of procedures, being elements of other structures;

7. As a particular case of the previous item, bulk type instances should have the same rights to persistence (which includes transience) as the other values in the language;
8. Explicit iteration (as well as the implicit iteration in the algebra) should be provided over the elements of a bulk type; in procedural languages, the order of iteration should be well defined, and it is desirable that it should be possible to specify it;
9. An interface between programming with bulk types and programming with the elements of that bulk type should be provided, for example, choosing an element from a set; and,
10. For the model of computation which the language supports (procedural, logic, transactional, concurrent, distributed, etc.) there should be reasonably efficient implementations of all the above features of bulk types.

2.2 Map types

The *map* proposed here is based on the mathematical notion of finite maps. In this abstract form they are a finite set of ordered pairs. The first element of the pair is called the *domain* and the second the *range*. When a value for the domain is supplied, the map returns the range. It is not always possible to apply the inverse operation.

These mathematical maps are like functions over finite domains. In our model, we are concerned with *finite* and *partial* maps. They are finite, because all the sets of pairs are finite. They are partial, because it is possible to supply the map with a value that is not in its domain. This finite and partial nature of maps makes them different from typical functions like *square*, but it is noted that an implementation of *square* in a typical language library is both finite and partial.

A more significant difference is that the maps we propose may be mutable, that is there are operations available on these maps that change their value while preserving their identity. There is an algebra defined over the maps which constructs new map instances. There are iterators and choice operations that allow code to operate on the elements within maps, utilising ordering when appropriate. Both mutation preserving identity, and iteration over the contained elements, are not usually defined for functions.

Since we are designing our maps to be integrated in the type systems of database programming languages, we first introduce map type expressions. Instead of using an abstract syntax, we follow the syntactic choices of Napier88 since we are currently experimenting with a prototypical implementation of maps in this language. Full details of our working definitions may be found in [ALPR91b] and details of implementation methods and their performance may be found in [ABL+91].

Map type expressions also allow constancy, equality and order to be specified. These additional properties of map types are discussed after the simpler map types have been illustrated (see Sections 2.2.1, 2.2.2, 2.2.3). When the range is omitted the map is required to behave exactly like a set. Some examples of map type declarations that will be used in this paper are given. The theme of our examples is the modelling of a bibliographic system. In appendix A, we collect all the examples used in this paper and we develop further the application.

```

type Itoi is map ( int → int )
type Stoi is map ( string → int )
                                !!Below a type to hold citation keys
                                !!a map from the keyType coded as an integer

```

```

                                !!to the string representing the key
type KeyMap is map ( int → string )

type Sstoi is map ( string, string → int )
type Authors is map ( int → Person )
type ItoAlpha [alpha] is map ( int → alpha )

type SetI is map ( int )
type Requirement is map ( string )
                                !!now the type to represent a reference
type Reference is map ( string → BibField )
type Requirements is map ( string → Requirement )
type Series is map ( string )
type Publisher is map ( Name, Address, Series )

```

In the examples above, we presume some other types (*Person*, *BibField*, *Name* or *Address*) have already been defined. We have defined a map that models a set of integers (*SetI*). The *Sstoi* map type has a multi-column domain and maps pairs of strings to integers. The *ItoAlpha* type describes map types which associate integers to instances of the type substituted for *alpha*, i.e. a dense or sparse one-dimensional array of elements of any type *alpha*. The reader should notice that our map types easily represent relations. A relation is a multi-column map with no range. We can also model NF^2 relations since there is no restriction on the types which appear as domains or ranges. For example, the *Publisher* map can be seen as a ternary NF^2 relation which associates a publisher's name, their address and their series set. We shall see in Section 3 that the algebra defined over maps allows the expression of database operations. Another way of modelling the publisher data using maps could have been the following:

```

type PublisherI is map ( Name → Address, Series )

```

The *PublisherI* type emphasises the access to the publisher information via the publisher's name and models the key constraint.

Most of the semantic data models [HK87] or of the specification models [Som89] use a "specification using functions" approach. Functional models have proved useful for modelling databases (see [HK87]) and specifications are often written using functions or mappings. A well known example of these approaches is the Z model [Spi88] where functions or mappings are sets of pairs where each pair shows how an output relates to an input. In Z, functions (or mappings) are used to model data structures such as data dictionaries. Our maps follow a philosophy similar to the Z functions. Since we have designed maps in order to be integrated with the type systems of database programming languages, we think that it will bring the advantages of these specification techniques directly into the programming constructs. On the other hand, we have designed this construct in order to allow manipulation and update by imperative programs.

2.2.1 Control of Map Mutability

Map types are not only a data modelling construct but also a programming construct. It is thus important to control the operations that are possible on the instances of a map type, and especially the mutability of these instances. Associated with a map type declaration is a map instance generator. Each instance has a unique and persistent identity, which is unchanged by operations on the map. Each map identity identifies a store holding that map's value. The allowed changes to the contents of that store may be restricted by constancy controls.

The syntax allows various specifications of the constancy at type declaration. As will be presented later, the range of a map may be altered by assignment, and the domain by insertion and removal. By default, all these operations on a map are allowed.

Preceding the word `map` by `constant` prevents any change of the domain after the map has been constructed. It is interesting to note here that an intermediary level of control could be provided. It would be possible to permit insertions, and forbid removals, e.g. **extensible map**, which would ensure that what the map previously mapped it would continue to map.

Instead of having constancy control over the whole range, it is provided for each element of the range. So it is possible to have some of the positions in the range updateable by assignment, and others disallowed on the left hand side of an assignment. Some examples follow.

```

    !!No additional constancy
type Publisher1 is map ( Name → Address, Series)
    !!Assignment to first column of range inhibited
type Publisher2 is map ( Name → constant Address, Series)
    !!Assignment to both columns inhibited
type Publisher3 is map ( Name → constant Address, constant Series)

```

Even if the whole range is designated constant as in *Publisher3* the range may still be changed by pairs of removes and inserts unless the domain has been made constant. One of the attractions of maps is the useful control, enforced via the type checker, that they give over various aspects of mutability.

2.2.2 Handling Equality in Maps

For a map to be well defined, equality must be provided for the elements of its domain. For example, consider a map $M: \text{map } SetI \rightarrow \dots$ with a domain type *SetI*, and an entry corresponding to *SetI* {1; 2}¹. One would expect it to be accessed by an application program using the expression $M(SetI\{1; 2\})$ ². However, this implies that the two expressions *SetI* {1; 2} are equal. This will not be the case if such a map creation expression generates a *new* database object, not equal (the default equality on composite objects is identity in Napier88) to the first. The intuitively desirable equality would have *SetI* {2; 1} also equal to these two sets.

Furthermore, it is inconceivable that an automatically generated version of equality will always match the user's intuitions. The only sensible solution is to allow a programmer to provide the equality, over-riding a default equality.

```

    !!a set of people for comparing author lists, etc.
type People is map ( Person with ( op = as samePerson))

```

The clause after `as` is required to return a value of type `proc (T, T → bool)` where T is the type of the domain of the map. Further discussions of the language design issues raised by maps appears in [ALPR91a].

2.2.3 Order and Efficiency

There is a need for a specification of order of processing in a procedural language. It is the ability to utilise such order *on appropriate occasions* that motivates such languages. The order of iteration over instances of maps is one such example. Several places occur where the order

¹See section 2.3 for this expression syntax.

²See section 3.1.1 for map access expressions.

could be defined: as a property of a type, as a property of an instance, and at iteration. The first is chosen, to make diadic operations easier to define and more efficient to implement. The syntax is similar to that given above for equality, the only change is that we allow the symbol “<”.

The default is the system defined definition of <. An internally defined ordering will be defined for mutable values that has no relationship to their contents. This will ensure that two iterations over the same map, which may have been updated, will take place in the same order, even if < has not been explicitly defined³.

```

                                !!The bibliography example continued
type Bibliography is map ( Reference with ( op = as sameRef, op < as alphaDate ))
type People is map ( Person with ( op = as samePerson, op < as alphabetic ))
type AuthorBib is map ( Pcrson with ( op = as samePerson, op < as alphabetic )
  → Bibliography)
type PersonCode map ( Person with ( op = as samePerson, op < as alphabetic )
  → constant int )

```

Some inequality operator is desirable if operations such as map look-up and union are to take place in an acceptable time. It means that internally maps can be kept as trees and diadic operations can be merges. The alternative of depending on hashing does not combine well with ordered iteration.

2.3 Creating map instances

This section presents the syntax for defining map instances. Expressions in the algebra over maps, presented in a later section (3.3), also generate map instances from existing ones.

It is believed that there should be a convenient notation for generating instances of any type in the language. This is one of the motivations for including a map constructor rather than attempting to build it out of primitives.

Some examples follow of the various forms of map instance generation.

```

let inits, fullInits, names, fullNames = 0, 1, 2, 3
let oldKeyCode = Stoi
  { “initials”      → inits
    “surnames”     → names }

let keyCode = Stoi
  { “fullInitials” → fullInits
    “fullSurnames” → fullNames
    “forMalcolm”  → inits  }

```

!!Now note initial requirements

```

let allReq = Requirement { “key”; “author”; “title”; “year” }
let requirements = Requirements {
  “article”      → allReq
  “book”        → allReq ++ Requirement { “publisher” }
  “inproceedings” → allReq ++ Requirement { “booktitle” } }

```

³It is necessary that the operations provided by a user for equality and ordering also satisfy the appropriate algebra, and are invariant over mutation. Failure to meet this requirement would be a programming error, and would cause *localised* failure of map operations. Whether this is a problem for programmers has to be investigated.

```

let mainBib = Bibliography { }

                                !!People can have a subset
type UserBib is map ( string → Bibliography )
                                !!We maintain a subset for each known author
let authorBibs = AuthorBib { }

                                !!And an index into author's, name,
                                !!initials, distinguishing code
let findAuthor = map ( string, string, int → Person ) { }
                                !!the code is 1 for the first person with the
                                !!initials and surname, 2 for the next, and so on
                                !!there are operations to merge and separate
                                !!people and to choose the sort order

                                !!Also fast access paths for keys
type IndexRef is map ( string → Reference )
let masterIndex = map ( int → IndexRef ) {
    inits           →      IndexRef { }
    fullInits      →      IndexRef { }
    names          →      IndexRef { }
    fullNames      →      IndexRef { } }
                                !!A translation table for Roman numbers
let roman         =      Stoi
{
    "I"           → 1
    "V"           → 5
    "X"           → 10
    "L"           → 50
    "C"           → 100
    "D"           → 500
    "M"           → 1000}

                                !!A translation table for English numbers
                                !!With French and German subtitles used for
                                !!producing French and German bibliographies
let nums         =
map (
    string      → int,      string,      string )
{
    "zero"      → 0,        "zero",   "nul"
    "one"       → 1,        "un",     "eins"
    "two"       → 2,        "deux",   "zwei"
    "three"     → 3,        "trois",  "drei"
    "four"      → 4,        "quatre", "vier"
    "five"      → 5,        "cinq",   "fünf"
    "six"       → 6,        "six",    "sechs"
    "seven"     → 7,        "sept",   "sieben"
    "eight"     → 8,        "huit",   "acht"
    "nine"      → 9,        "neuf",   "neun"
    "ten"       → 10,       "dix",    "zehn"
    "eleven"    → 11,       "onze",   "elf" }

```

```

                                !!A publisher table
let publishers = Publisher
  {"Addison Wesley", "USA", Series{"Frontier"
                                   "Anthology"
                                   "Tutorial"
                                   "History" }}

```

The extensive set of examples above illustrate some of the variety of maps that may be obtained, and easily written as expressions, which can be read and understood. They will be used in the example operations that follow.

3 Map Access and Manipulation

3.1 Maps as Mutable Objects

As was mentioned above, there are three ways in which a map may be altered without changing its identity. That is, a map behaves as a store, and any references to it, held in other stores, remain valid over these operations, so that data accessed via such references are the new values. The three operations are:

1. Assignments to components of a range;
2. Insertions of new maps; and,
3. Removal of parts of the map.

Each of these operators will be dealt with in turn. The assignment depends on the mechanism for naming a row of the table, which may be used in both the right and left context. All mutation operations are atomic.

3.1.1 Access to table elements

There are three ways of accessing individual elements of maps:

1. By a map access expression;
2. By iteration (see section 3.2); or,
3. By a map choice (see section 3.2.2).

The access expression is analogous both to vector (array) indexing, and to function application. It returns the range components of the selected row. It may generate an error if there is no match in the domain. When an access expression is used anywhere other than to the immediate left of an assignment operator, it is analogous to a procedure application. Some examples using the previous map values illustrate the right and left context forms.

```

keyCode ( "forMalcolm" ) := fullInits
                                !!copy Paul's choice
keyCode ( "forMalcolm" ) := keyCode ( "forPaul" )

```

In the right context, where the range is unary, the behaviour of the access expression is similar to that of a function. Actual parameters are provided, and a result value is returned. In such cases, the map is searched to find a row where the values in the key match the stored domain values, since the domain is a set there will be one or zero such rows. If there is a row, then the value in the range is returned. If there is no row, then the system generates an error event, which may be converted into an exception. The matching of values uses the equality defined for this map. Some examples using bindings follow:

```

                                !!Save the translation
let i, f, g := nums ( "four" )
                                !!Sabotage 4
nums ( "four" ) := 22, "vingt-deux", "zwei-und-zwanzig"
                                !!And 11
nums ( "eleven" ) := -, -, "zwölf"
                                !!Then prove we're nice guys — tidy up
nums ( "four" ) := i, f, g
nums ( "eleven" ) := -, -, "elf"

```

The don't care symbol (`_`) matches any type, and substitutes for the position of a clause. It means that the corresponding name is unchanged, and is required when that name is in a column constrained to be constant. Assignment can be fast, as it cannot violate the set constraint on domains, and hence no check is necessary on assignment.

3.1.2 Insertion

The expression before `insert` and that after it must yield maps of exactly the same type except that the constancy of the second map is ignored. The insertion operation adds additional rows to the stored map, while respecting the requirement that the domain remains a set at all times. The `insert` clause does not produce a result, i.e. the result is `void`. Some examples follow:

```

                                !!insert an old keyCode map into the current one
keyCode insert oldKeyCode

                                !!insert two new values into keyCode
keyCode insert Stoi
  {
    "forPhilippe" → inits
    "forMalcolm" → fullNames
    "forChris"   → inits
  }

```

The clause preceding the `insert` must evaluate to a map, which becomes the map that has its value changed. The map yielded by the clause after the `insert` is left unchanged. The new value in the first map is the result of adding new rows if the domain value is not already present in the map, in the example "forPhilippe" and "forChris" and of leaving unchanged rows whose domain value matches a domain value of the `insert` clause ("forMalcolm" in the example). In the case where `m=0` (no range), the maps behave as sets, and this is set union.

3.1.3 Removal

The clauses before and after `remove` must have domains of exactly the same type, but the second map may have a constant domain and its range may be a (possibly empty) subset of the columns of the first map. `remove` removes from the first map all the bindings whose domain value matches a domain value in the map yielded by the second map. Additional entries in the second map have no effect. Some examples are given:

```

                                !!Restore keyCode to its pristine state
keyCode remove Set[ string ]
  {
    "forChris"
    "forPhilippe" }

```

3.1.4 Replacement

Subject to the constraints of constancy, it is also possible to replace the ranges of several entries in the map simultaneously. An example follows:

```

                                !!Change the interpretation of key codes
keyCode replace Stoi
  {
    "forMalcolm" → fullnits
    "forPhilippe" → fullnits
    "forChris"   → fullnits }

```

This clause changes the ranges of those entries that lie in the intersection of both maps, so that in the first map, they now map to the values they mapped to in the second map. This provides a succinct notation for the atomic application of a set of accumulated changes. It may also be easy to implement this so that it is considerably more efficient than the equivalent sequence of individual assignments.

3.2 Iteration and Choice

A critical issue in the provision of types is how transfers are made from the bulk type to elements of the bulk type. For example, given a singleton set, how is the element of that set obtained?

Two categories of transfers from bulk type to element are presented:

1. Iterations, which arrange that variables introduced in the iteration, take on all the values in the map in the defined sort order; and,
2. Choices, which arrange that similar variables take on the value of one entry in the map.

These two depend on a common binding mechanism which is also used in the algebra. Identifiers may be introduced which denote (some of) the elements in a row. They denote the R-value in the case of the domain and constant range elements, and the L-value for the remaining range elements, thus enforcing constancy requirements, since compilers prohibit the use of R-values immediately to the left of assignment operators. This is similar to the use of identifiers in pattern matching and in comprehensions [Tri91].

The context in which a binding appears will statically determine the type M of the maps to which binding may take place, and hence statically determine the types of the introduced element identifiers. For each lexical occurrence of a binding, one map type, say M , will be the subject of the binding. An example of binding is:

```
english → -, -, german
```

This binds a new variable *english* to the first element of the domain, giving it the type of the first column of the domain of M . Similarly, the type of *german* will be that of the third column of the range of M . In this example, *english* is constant, and *german* will be a variable unless M has that column specified constant.

Each execution of the binding will associate it with exactly one row of the map, and the introduced identifiers then denote the corresponding elements of that row.

3.2.1 Iteration over Maps

This arranges that the specified binding takes on the values of every entry in the map, in the ordering specified in the type. An example follows:

```

                                !!Print a translation table
for each english → -, -, german in nums do
begin
    newline (); writeString ( english ); tab (); writeString ( german )
end

                                !!Collect even numbers
let evenNums = map (string → int ) { }
for each s → i, -, - in nums where even(i) do evenNums insert s → i

                                !!Increment all numbers
for each →i in nums do i := i + 1

```

If the **where** clause is omitted, the iteration is performed for every entry in the map, the clause after the **do** being executed once per entry. If the **where** clause is present, the iteration is executed as if each entry were visited in turn, but only for those entries for which the **where** clause evaluates to **true** is the clause following the **do** evaluated. Compilers should optimise the use of **where** clauses.

3.2.2 Choice from Maps

All such choices are introduced by one of two system words **use** or **removing**. They differ in that **use** does not, itself, change the map, whereas, **removing** removes the entry after the binding has been extracted. It gives a simple way of making it clear that a program is terminating over a finite map, and that it is deterministic if it is eroding the set, as it cannot be perturbed by (potentially concurrent) updates to the set occurring between choice and removal. It is often the case that the **removing** form will halve the number of accesses into a map.

The **use** (or henceforth **removing**) introduces a clause which when evaluated is the map to which all the bindings will be made. This is followed by **with** introducing a list of bindings, each of which introduces new variables into scope as before. If all bindings succeed, then that clause is evaluated once in the context of all those bindings. The interpretation of each of the choice options is:

- some** The system chooses any entry in the map (satisfied by the **where** clause if present). This matches the set axiom of choice. The majority of implementations will choose the first entry, but this is not guaranteed.
- first** This chooses the first entry in the map according to its sort order defined by the map's type. Programmers are invited to hold the intuition that this choice is fast. If the **where** clause is present, this may not be fast, depending on the complexity of that clause, but the binding is then to the first entry that satisfies the clause.
- last** As for **first**, but this time the last entry in the stored order is used.
- the** This first verifies that the map, after filtering by the **where** clause if present, is a singleton map. If it is not, then an error event occurs. If it is, then the one entry in the map is used.

An error event occurs if the map is empty, or if it appears empty after filtering through any **where** clause present. The use of a **where** clause in this context cannot easily be simulated by other language features. Two examples follow:

```
use nums with last → i, _ in
begin
  newline (); writeString ( "The last named number is " )
  writeInt ( i )
end
```

```
use nums with the → i, s, _ where i = 12 in
begin
  newline (); writeString ( "The French word for 12 is " )
  writeString ( s )
end
```

3.3 An Algebra over Maps

This section defines several operations over maps, which given one or more maps as arguments, leave those arguments *unchanged*, and *generate a new map* as a result. There are four forms: union, intersection, difference and generation. The first three have their meaning inherited from set algebras. Generation is similar to set comprehension in its power, but we hope that it is easier to read and understand.

3.3.1 Union, Intersection and Difference

The union operation is represented by **++**, the intersection operation by ****** and the difference operation by **--**. Both operands are required to be of closely related types, and the result has the same type as the first operand. The constancy of the second operand is ignored when checking type compliance. All three behave as set operations with the usual algebraic properties when the range is empty. All three may be implemented by merge algorithms, without a sort, as the type rules guarantee their two arguments have the same order.

3.3.2 Union

Both operands are required to be of exactly the same type, and the result is also of that type. The union operation forms a new map that maps all the values that the first argument mapped, and all the values that the second argument mapped. Note the same definition of union was used for **insert**, but in that case the result was constructed in the first argument. An example follows:

```
!!Collect in a new map some English numbers
!!We assume that oldNums is another instance of
!!the nums type map
let newNums = nums ++ oldNums
```

3.4 Symmetry and Contradictions

It is desirable that intersection and union retain their symmetric properties, so that a more extensive set of optimisations of generators is available, and so that they match the intuitions of programmers evoked by their names. Consequently, if there is a non-empty intersection between the two maps, it must be treated symmetrically. This is trivially satisfied if the same domain values map to the same range values. However, where they contradict, i.e. the same range values map to different values in the two maps, an error event is generated. This preserves symmetry and optimisability, and is used in the definitions of **insert**, **remove**, union, intersection and difference. In the cases of **remove** and difference, only the range columns common to both maps are considered.

3.4.1 Intersection

The intersection, ******, has the type of the first operand and is obtained by keeping all the values of the first map whose domain value matches a domain value in the second map, and whose range values do not contradict.

```

                                !! Collect person codes of registered authors
!! personCodes and authorBibs are instances of maps PersonCode and AuthorBib
let registredCodes = personCodes ** authorBibs

```

3.4.2 Difference

The difference is represented by **-**. The operands are required to have the same domain types but the columns of the range of the second map may be a subset of those in the first map. The result has the type of the first operand and is obtained by keeping all the values of the first map whose domain value does not match a domain value in the second map. An example follows.

```

                                !! Collect person codes of non-registered authors
!! personCodes and authorBibs are instances of maps PersonCode and AuthorBib
let nonRegisteredCodes = personCodes --authorBibs

```

Using the above operations, one can easily build more complex operations such as the overriding operator of Z . Suppose, we want to build a map *newNums* from *nums* and *oldNums* which contains all values of *nums* which are not in *oldNums* and replace the values of *nums* for which a domain value matches a domain value in *oldNums* by the entry in *oldNums*. This is done as follows.

```

let newNums = (nums --(nums ** oldNums)) ++ oldNums

```

3.4.3 Generation

The generation construct has the power to construct derived values from one or more maps. In fact, it is intended to have the expressive power of selections, projections and theta-joins [Cod79], and of comprehensions [Tri91]. No provision is made for recursive generators, as programs are more legible if the programmer uses procedural recursion explicitly (contrast with DBPL [MS89]). Generation expressions *do not* define the order in which the result is constructed, in order to leave freedom for optimisers. It is a matter for investigation whether this will then allow the construct to be compiled via the comprehension techniques [TCD90] or their equivalent. Preliminary work on optimisations is reported elsewhere [ABL⁺91].

Generation expressions require the type of the result to be specified. This allows projection, change of constancy, change of stored order, and change of equality. It is made mandatory so that there are no complex rules about the default type that have to be implemented in the compiler and understood by the programmer. Several examples now follow, to illustrate various uses of these generators.

```

                                !!An example of map inversion
let codePersons = constant map (int → constant Person)
                                { each p → i in personCodes : i → p }

                                !!Examples of projection to maps
let gNums = Stoi { each → i, -, gs in nums : gs → i }
                                !!and combining filters via set operations
let knownNums = Stoi {(each es → i, -, _ in nums : es → i) ++
                                (each → i, fs, _ in nums : fs → i) ++
                                (each → i, -, gs in nums : gs → i) }

                                !!The above expression may be optimised to a merge
                                !!An example of selection and projection to a set
let addisonSeries = map (Name, Series)
                                { each n, -, s in publishers where n = "Addison Wesley" : n, s }
                                !!An example of an equijoin
let romanEnums = map(string → string)
                                { each rs → ri in roman ;
                                each es → ci, -, _ in nums where ri = ci
                                : rs → es }

                                !!An example of changing the order
let lastAreFirst = constant map(Person with ( op < as
                                proc ( a, b: Person → bool); ... → constant int)
                                { each p → i in personCodes : p → i }

```

The clauses evaluating to maps that appear after **in** are evaluated from left to right, in the usual fashion. However, expressions in their **where** clauses, and in the map row expression of the target (after each colon **:**) are evaluated in an unspecified order. The programmer is therefore strongly advised not to use procedures in these contexts which have side-effects, and certainly not to depend on the order of execution of those side-effects. This is the one place where there is no commitment to a specified order of execution to permit the full use of optimisation techniques.

The expressions in the map row are evaluated once per element in the cartesian product of the elements produced by the filters, in arbitrary order. If the result of such an evaluation produces a row with the same domain, but a different range value to one previously generated in this generation expression, then an error event occurs.

Since our maps can represent relations, one can see that the operations presented above emulate the algebra, and we expect them to be susceptible to similar optimisation technology. However, making equality and order part of the map's type means that certain additional optimisations, e.g. recognising when to use a merge algorithm, can sometimes be performed statically.

3.5 Monadic operators

Two monadic operators are provided. The filter operation, such as:

```
each  $x, y \rightarrow z$  where predicate(  $x, y, z$  ) in aMap
```

constructs a subset of the map after **in**, defined by the predicate after **where**. The predicate is called in map order, so that predicates with side-effects, e.g. which interactively consult the user, may be used.

The **copy** operation is equivalent to filter where the predicate always yields true. For both these operators, the result is of the same type as the input map.

4 Data Modelling with Maps

The preceding section introduced maps from the programming viewpoint. The need for bulk constructs and bulk manipulation has been well understood since the pioneering work of Pascal-R [Sch77]. The subsequent development represents a two pronged search; on the one hand for more powerful and general bulk operations, and on the other hand for more modelling power. The first search is constrained by limits to what programmers will readily understand and by limits to the complexity of implementation techniques that may be deployed. Consequently, maps stop short of the complexity of DBPL's operations (e.g. transitive closure operations are not denoted by recursion and automatically provided in maps, and the context of optimisation is both limited and explicit). The operations on maps are intended to present a set of high-level and general purpose *primitives* to programmers, from which they may build more sophisticated structures. Programmers are recognised as being able to contribute to performance by using the primitives wisely.

As data type complete persistent value constructors, the map construct has considerable modelling power. As already remarked, it has the power of relations, but being data type complete, this includes all NF^2 relations, and it includes sets. It also has the power to model asymmetric or directed relationships. This is particularly important where values include identities, as in all object-oriented systems, since it describes dependency relationships. The definition of types is considered precisely equivalent to the definition of schemata.

The modelling capabilities of maps will be illustrated using two familiar examples: the parts and suppliers database introduced by Date [Dat77] and the parts explosion [AB87]. These examples utilise the Napier88 syntax, and some of the other Napier88 type constructors [MBC⁺90].

For the parts example exactly Date's model may be specified by:

```
type Parts is map ( P#, Name, ... )
type Suppliers is map ( S#, Name, Address, ... )
type Supplies is map ( P#, S#, ... )
```

However, using maps other models may be specified. For example, we can model a part and a supplier separately, giving them identity, and form a model equivalent to that above.

```
type Part is ...
type Supplier is ...
!!Sct[ $\alpha$ ] was defined in terms of maps above
type Suppliers is Sct[ Supplier ]
type Supplies is map ( Part, Supplier )
```

This demonstrates that it is possible to impose and name more structure, while retaining the symmetrical treatment traditional in relations. It may be, that an asymmetric model of this

information is required. For example, if it is known that computations may require suppliers of a given part, but never require parts supplied given a supplier, the following two definitions might be considered:

!!Definition 1 — independent suppliers

```

type Part is ...
type Supplier is ...
type Parts is Set[ Part ]
type Suppliers is Set[ Supplier ]
type Supplies is map ( Part → Supplier )

```

!!Definition 2 — dependent suppliers

```

type Supplier is ... )
type Part is structure (
  PartNumber: int;
  PartName: string;
  SuppliedBy: Set[ Supplier ];
  ... )
type Parts is Set[ Part ]

```

In the first case, if a part is no longer used by the company, the programmer not only has to dispose of the part, but also of all related entries in Supplies. In the second case, this is not necessary, and as the Supplier set has not been constructed, a supplier that no longer supplies parts is no longer remembered. These are just examples of the many nuances that can be modelled using maps.

Only one of the many possible solutions to the parts example will be given. (In fact, the structure that has been used to run the Sun benchmark [CS90] against maps.)

```

type Supplier is structure ( Name: string; ... )
!!quick access by name
rec type BasePart is structure ( Cost: int; Mass: real;
  SuppliedBy: Set[ Supplier ] )
& CompositePart is structure ( AssemblyCost: int; MassIncrement: real;
!!range denotes the quantity
)
& BaseOrComp is variant ( Base: BasePart; Comp: CompositePart )
& Part is structure ( Name: string; Number: int;
  UsedIn: Set[ CompositePart ];
  Specialisation: BaseOrComp )
type PartByName is map ( string → Part )
type PartByNumber is map ( int → Part )

```

This solution avoids join-like processing for the main transitive closure calculations, such as: what is this part made from, in what assemblies is this part used, what is the cost and mass of this part, what suppliers supply components for this part, etc. Note that many maps of each type may be constructed, for example:

!!Variables are initialised as they are declared

!!PartsDB is an environment made persistent

```

in PartsDb let suppliedParts := PartByName{ }
in PartsDb let availableParts := PartByName{ }
in PartsDb let partsInRAndD := PartByNumber{ }
in PartsDb let discontinuedParts := PartByName{ }

```

5 Related Work

Maps have been presented as a construct for manipulating bulk data in database programming languages and as an expressive data modelling construct. Every database programming language or object-oriented database system can thus be compared with the maps approach with respect to bulk data manipulations. Some of these languages provide only one constructor for bulk data, mostly some kind of sequence or array constructor (sequences in Galileo[ACO85], sets in DBPL[EEK+85], ...), and some languages provide several unrelated constructors (sets and lists in O₂ [CDLR90], vectors and images in Napier88 [MBC+88]). In [ALPR91b] there is further discussion of the representation of such language constructs in terms of maps.

A survey paper identified the utility of bulk types [AB87]. Proposals for bulk types are now a topic of much interest [MS91, BTBN91, ALPR91a, Tri91, RS91b, RS91a, SP91, GV91]. Implementations and evaluations for the intended range of applications are necessary before it will be possible to resolve the relative merits of these proposals, including maps.

Colleagues at Glasgow have developed a categorisation of a class of bulk types [TW91]. Proposals by FIDE colleagues in Pisa have developed a class of bulk types to represent relationships in Object-Oriented systems [AGO91]. These are similar to the the following variation of definition 1 given above:

!!Definition 1 revised to model Nuovo Galileo

```

type Part is ...
type Supplier is ...
!!like their extents
type Suppliers is Set[ Supplier ]
!!like their relationships

```

The COMANDOS group also have a similar construct [HN91]. It is hoped that maps will prove an appropriate set of implementation primitives to support such data modelling constructs with additional behavioural extents.

Maps are not new. The functional data model of Daplex [Shi81] and its antecedents and developments was constructed out of finite maps. However, they were not defined with the same parameters, algebra and ordering. Similarly Z utilises finite maps, but without the corresponding implementations [Spi88]. MUMPS was an early (1967) use of maps for database applications in health care [Con83]. Although MUMPS provides no algebra, and has a very limited parameterisation of maps, it is a demonstration that programmers using Basic have been able to understand the construct, and that implementation for moderate quantities of data is feasible.

5.1 The DBPL database programming language

A comparison with DBPL is appropriate, as it is the culmination of a long programme of research into bulk types in DBPLs. The DBPL database programming language has been developed in the university of Hamburg as a successor to the Pascal-R language. It is a fully upward compatible extension of the Modula-2 language and its main feature is a flexible set construction. DBPL also inherits an array construction from Modula-2.

A DBPL set is a collection of elements of the same type with no duplicate elements in it. A set can be keyed, the key introducing the additional constraint that the set cannot contain two elements having the same key. A relation can be trivially represented as a keyed set of flat tuples. The key then provides access like the access using the domain provided by maps.

DBPL sets can be used to represent maps in the following way : the domain and range elements of the map type are gathered into a DBPL record structure and a set is constructed with the domain elements of the record as a key. We have, for example :

```

type  NumsRecord = record
    domain_1 : string,
    range_1  : integer,
    range_2  : string,
    range_3  : string
end
NumsType = relation domain_1 of NumsRecord

```

In maps names are introduced via positional bindings when required, whereas, in DBPL the columns are named. Maps have a retained and usable stored order, which may be specified, whereas, the ordering in which the elements of a DBPL set are accessed cannot be specified and may not be deterministic. This increases the opportunity for optimisation, but requires that an intermediate structure be used to hold results before a sorted version can be used. Consider printing a telephone directory, or bank statements.

Set manipulations in DBPL have commonalities with our map algebra. Indeed, sets have the classical set operations like intersection, union, and difference. Set elements can be selected using **selector** expressions and new sets can be constructed from existing sets using **constructor** expressions. DBPL constructors are formally more powerful than map generators as they directly specify recursive data traversal. Whereas, with maps the programmer must program such recursion explicitly. This reflects the more primitive nature of maps, a choice that makes them more suitable as a store interface structure. It may also avoid the problems of optimising a more general algebra encountered in DBPL.

It has not proved possible to make DBPL sets data type complete, although transient sets may contain reference types these may not persist.

More generally, the DBPL language has a calculus style, in contrast we propose an algebraic style for manipulating maps.

6 Conclusions and Directions of Research

The type constructor **map** has been defined as a highly parametric type which can be used to represent most bulk types found in database systems or programming languages. The use of a number of parameters rather than a variety of constructors is more powerful. A greater spectrum of instance forms can be described and constructed, and these can be combined with simpler and more regular rules, than would be the case with a model or language that depended on a number of less parametric constructors.

The **map** construct is a powerful modelling concept capable of providing the functionality of sets, relations, and the relationships in functional data models. They have been demonstrated, in [ALPR91b], as capable of serving as replacements for: vectors, arrays, sparse arrays, indexes, sets, and as representations of relationships. They have a means of defining order, so that their construction can perform a sort, and their sorted form can be exploited during iteration.

The operators over maps provide: map construction expressions; update in-situ; an algebra for deriving new maps from existing maps with a power similar to that of relational query languages or set comprehensions; iteration and individual access to components. They are designed to behave as sets when their range is empty, and as vectors or arrays when their domain is dense. They integrate well with existing imperative programming languages. When

parameterised with degenerate forms of equality and ordering they may also model sequences and bags.

It is believed that with this combination of modelling power and operations the maps will satisfy many of the requirements for bulk types. Two uses are envisaged, as a bulk type in persistent programming languages, and in abstract machines as an interface type to a canonical form in persistent object stores. It is mainly the first use that has been explored in this paper. The details of a map construct have been explored for the programming language Napier88. It is expected, however, that most aspects of the design would carry over to any strongly typed language.

6.1 Current Experiments

There are two implementations of map stores operational, with procedural interfaces, and adaptive persistent representations. One is written in C++, the other in Napier88. They are the subject of experiment and measurement. Early results are reported in [ABL⁺91]. That paper also reports on the initial analysis of optimisations for maps.

A prototypical set of extensions, providing the language *maps*, using the notations of this paper is being developed from the Napier88 compiler. This will provide a vehicle for evaluating programmer reaction in the near future. Meanwhile a few dedicated users depend on the procedural interface.

6.2 Research Directions

Because maps adequately represent and denote many existing language constructs, an entirely new language should be designed, utilising the potential for simplification that maps offer. Such a design is underway, and is intended to lead to the language Maxwell. That language would then enable a more thorough evaluation of programmer behaviour when provided with maps. However, many issues arise in the design of such a language [ALPR91a].

The use of maps to supplant or support other data models and constructs, such as object-oriented models, should be explored. As maps offer many additional representations of an organisation, the appropriate design methodology that leads to the optimum maps will have to be developed.

An abstract machine that includes operations on maps is required. Compilation and optimisation techniques will be developed to exploit this abstract machine from map languages. For example, optimisations may use ordering, indexing, clustering, merging and re-arrangement of operation order.

A persistent object store supporting concurrent transactional use of these map types should be designed and evaluated. This requires a model of concurrent and transactional use of maps. It should be noted that all the update in-situ operations on maps are defined to be atomic, and that this is believed to help with these design issues, as it increases the granularity of the interaction with the store.

These are just three aspects of a more general investigation that could be carried out using maps. Matties and Schmidt have reported a preference for "added-on" rather than "built-in" bulk types [MS91]. That is a particular position on a continuum. The next step is to build a modified compiler leaving the abstract machine unchanged. Changing the abstract machine but leaving the store intact is a further step. These adaptations can continue as far as the hardware components themselves. Characterising how far to go with building in primitives at all such architectural levels is an important investigation for bulk types. Maps are put forward as one possible bulk type worthy of investigation at a number of levels.

7 Acknowledgements

The authors wish to acknowledge the support of the European Community ESPRIT Basic Research Action 3070. Malcolm Atkinson also benefited from the hospitality of GIP Altaïr, from a fellowship given by the French government, and from the British SERC grant No. GR/F289543.

The discussions among the entire FIDE community, and particularly at the Types Group meetings were invaluable in developing and testing our ideas.

References

- [AB87] M.P. Atkinson and O.P. Buneman. Types and persistence in database programming languages. *ACM Surveys*, 19(2):105–190, June 1987.
- [ABL+91] M. Atkinson, V. Benzaken, C. Lécluse, P. Philbrow, and P. Richard. Experiments with Persistent Map Stores, June 1991. Submitted for publication.
- [ACO85] A. Albano, L. Cardelli, and R. Orsini. Galileo: A Strongly Typed, Interactive Conceptual Language. *ACM Transactions on Database Systems*, 10(2):230–260, 1985.
- [AGO91] A. Albano, G. Ghelli, and R. Orsini. A relationship mechanism for a strongly typed object-oriented database programming language. In *Proceedings of the 17th International Conference on Very Large Data Bases (3rd-6th September 1991, Barcelona, Catalonia, Spain)*, 1991. To appear.
- [ALPR91a] M.P. Atkinson, C. Lécluse, P. Philbrow, and P. Richard. Design issues in a map language. In *Proceedings of the Third International Workshop on Database Programming Languages (Nafplion, Greece, 27th-30th August 1991)*, 1991. To appear.
- [ALPR91b] M.P. Atkinson, C. Lécluse, P. Philbrow, and P. Richard. Maps as a type-quark. Technical report, Department of Computing Science, University of Glasgow, Glasgow G12 8 QQ, Scotland, 1991. FIDE report in preparation.
- [ART91] M.P. Atkinson, P. Richard, and P. Trinder. Bulk Types for Large Scale Programming. In J. W. Schmidt, editor, *Information Systems for the 90s: Workshop in Kiev October 90*, Berlin, Germany, 1991. Springer-Verlag.
- [BTBN91] V. Breazu-Tannen, O.P. Buneman, and S Naqvi. Structural recursion as a query language. In P. Kanellakis and J.W. Schmidt, editors, *Proceedings of the Third International Workshop on Database Programming Languages (Nafplion, Greece, 27th-30th August, 1991)*, 2929 Campus Drive, Suite 260, San Mateo, Calif. 94403, August 1991. Morgan Kaufmann Publishers. To be published.
- [CDLR90] S. Cluet, C. Delobel, C. Lécluse, and P. Richard. Reloop, an Algebra Based Query Language for an Object-Oriented Database System. *Journal of Data and Knowledge Engineering*, 5:333–352, 1990.
- [Cod79] E.F. Codd. Extending the relational model of data to capture more meaning. *ACM Transactions on Database Systems*, 4(4):397–434, December 1979.
- [Con83] M. Conway. *ANS MUMPS Programmer's Reference Manual*. MUMPS Users Group, 1983.

- [CS90] R.G.G. Cattell and J. Skeen. Engineering database benchmark, March 1990. Database Engineering Group, Sun Microsystems.
- [Dat77] C.J. Date. *An Introduction to Database Systems*. Addison-Wesley Publishing Company, second edition, 1977.
- [DCBM89] A. Dearle, R. Connor, A.L. Brown, and R. Morrison. Napier88 - A Database Programming Language? In *Proceedings of the Second International Workshop on Database Programming Languages (Oregon, June 1989)*, pages 213-230, 1989.
- [EEK+85] H. Eckhardt, J. Edelmann, J. Koch, M. Mall, and J.W. Schmidt. Draft Report on the Database Programming Language, DBPL. Technical report, Johann Wolfgang Goethe - University, Frankfurt am Main, West Germany, 1985.
- [GV91] Grumbach and Vianu. Expressiveness and complexity of restricted languages for complex objects. In P. Kanellakis and J.W. Schmidt, editors, *Proceedings of the Third International Workshop on Database Programming Languages (Nafplion, Greece, 27th-30th August, 1991)*, 2929 Campus Drive, Suite 260, San Mateo, Calif. 94403, August 1991. Morgan Kaufmann Publishers. To be published.
- [HK87] R. Hull and R. King. Semantic Database Modeling: Survey, Applications and Research Issues. *Theoretical Computer Science*, 19(3), September 1987.
- [HN91] D. Harper and M. Norrie. Data management for object-oriented systems. In *Proceedings of the 9th British National Conference on Databases (10th-12th July 1991, Wolverhampton, England)*, 1991. To appear.
- [Lam86] L. Lamport. *LaTeX user's guide and reference manual*. Addison-Wesley Publishing Company, Reading, Ma., USA, 1986.
- [LR89] C. Lécluse and P. Richard. The O_2 Database Programming Language. In *Proceedings of the Fourteenth International Conference on Very Large Data Bases*, August 1989.
- [MBC+88] R. Morrison, A.L. Brown, R. Carrick, R.C. Connor, and A. Dearle. The Napier88 Reference Manual. Technical Report PPRR-77, Universities of Glasgow and St Andrews, 1988.
- [MBC+90] R. Morrison, A. Brown, R. Carrick, R. Connor, A. Dearle, and M.P. Atkinson. The Napier Type System. In J. Rosenberg and D. Koch, editors, *Proceedings of the Third International Workshop on Persistent Object Systems: Their Design, Implementation and Use (Newcastle, N.S.W., Australia, January 1989)*. Springer-Verlag, 1990.
- [MS89] F. Matthes and J.W. Schmidt. The Type System of DBPL. In R. Hull, R. Morrison, and D. Stemple, editors, *Proceedings of the Second International Workshop on Database Programming Languages (Oregon, June 1989)*, pages 219-225, San Mateo, Calif., USA, 1989. Morgan Kaufmann Publishers.
- [MS91] F. Matthes and J.W. Schmidt. Bulk Types: Built-In or Add-On? In *Proceedings of the Third International Workshop on Database Programming Languages (Nafplion, Greece, 27th-30th August 1991)*, 1991. To appear.
- [PM88] J. Peckham and F. Maryanski. Semantic Data Model. *ACM Computing Surveys*, 20(3), September 1988.

- [RS91a] J. Richardson and Schwarz. Mdm: An object-oriented data model. In P. Kanellakis and J.W. Schmidt, editors, *Proceedings of the Third International Workshop on Database Programming Languages (Nafplion, Greece, 27th-30th August, 1991)*, 2929 Campus Drive, Suite 260, San Mateo, Calif. 94403, August 1991. Morgan Kaufmann Publishers. To be published.
- [RS91b] Rosen and Shasha. Rationale and design of bulk. In P. Kanellakis and J.W. Schmidt, editors, *Proceedings of the Third International Workshop on Database Programming Languages (Nafplion, Greece, 27th-30th August, 1991)*, 2929 Campus Drive, Suite 260, San Mateo, Calif. 94403, August 1991. Morgan Kaufmann Publishers. To be published.
- [Sch77] J.W. Schmidt. Some High Level Language Constructs for Data of Type Relation. *ACM Transactions on Database Systems*, 2(3):247-261, September 1977.
- [Shi81] D.W. Shipman. The functional data model and the data language DAPLEX. *ACM Transactions on Database Systems*, 6(1):140-173, March 1981.
- [Som89] I. Sommerville. *Software Engineering*. Addison-Wesley Publishing Company, third edition, 1989.
- [SP91] Small and Poulouvasilis. An overview of pfl. In P. Kanellakis and J.W. Schmidt, editors, *Proceedings of the Third International Workshop on Database Programming Languages (Nafplion, Greece, 27th-30th August, 1991)*, 2929 Campus Drive, Suite 260, San Mateo, Calif. 94403, August 1991. Morgan Kaufmann Publishers. To be published.
- [Spi88] J. M. Spivey. Understanding Z. A Specification Language and its Formal Semantics. In *Cambridge tracts in theoretical computer science*, number 3. Cambridge University Press, 1988.
- [TCD90] P.W. Trinder, D.K.C. Chan, and Harper D.J. Improving Comprehension Queries in PS-algol. In R.L. Cooper, A. Stewart, and P.W. Trinder, editors, *Proceedings of the 1990 Glasgow Database Workshop*, pages 103-120, Glasgow G12 8QQ, Scotland, March 1990. Department of Computing Science, University of Glasgow.
- [Tri91] P. Trinder. Comprehensions, a query notation for dbpls. In *Proceedings of Third International Workshop on Database Programming Languages (Nafplion, Greece, 27th-30th August 1991)*, 1991.
- [TW91] P. Trinder and D.A. Watt. Towards a theory of bulk types, 1991.

A The bibliographic Example

We develop in this appendix fragments of data definition and associated code for a bibliographic system which regroups most of the examples we use throughout this paper. It is assumed that data will be held to describe a collection of references that some group wishes to be able to cite. Each reference is held in a form that is related to a bibTeX representation [Lam86]. There are certain required fields of every reference that are held as fields of a record or structure in a traditional implementation. Here they will be represented as a map from strings to a variant type. The variant type is declared first, and explained as it is developed.

```

!!Subtypes corresponding to special fields
!!the remaining fields are held as strings

!!First a type to hold citation keys
!!a map from the keyType coded as an integer
!!to the string representing the key
type KeyMap is map ( int → string )
!!with here an initial value of map codings
let inits, fullInits, names, fullNames = 0, 1, 2, 3
let keyCode = Stoi
  { "initials" → inits
    "fullInitials" → fullInits
    "surnames" → names
    "fullSurnames" → fullNames
    "forMalcolm" → inits }

!!a sequence of type Person for authors and editors
type Authors is map ( int → Person )
!!a set of people for comparing author lists, etc
type People is map ( Person with ( op = as samePerson, op < as alphabetic ) )

!!and so the variant
type BibField is Variant (
  bibType: string
  keys: KeyMap
  authors: Authors
  title: structure ( tit: string, interpret: bool )
  year: int
  month: int
  other: string )

!!now the type to represent a reference
type Reference is map ( string → BibField )

!!and now the minimum requirements for bib types
type Requirement is map ( string )
type Requirements is map ( string → Requirement )

!!Now note initial requirements
let allReq = Requirement { "key"; "author"; "title"; "year" }
let requirements = Requirements {
  "article" → allReq
  "book" → allReq ++ Requirement { "publisher" }
  "inproceedings" → allReq ++ Requirement { "booktitle" } }

!!The bibliography
type Bibliography is map ( Reference with ( op = as sameRef, op < as alphaDate ) )
let mainBib = Bibliography { }

!!People can have a subset

```

```

type UserBib is map ( string → Bibliography )
                                !!We maintain a subset for each known author
type AuthorBib is map ( Person with ( op = as samePerson, op < as alphabetic )
                                → Bibliography )
let authorBibs = AuthorBib { }

                                !!And an index into author's, name,
                                !!initials, distinguishing code
let findAuthor = map ( string, string, int → Person ) { }
                                !!the code is 1 for the first person with the
                                !!initials and surname, 2 for the next, and so on
                                !!there are operations to merge and separate people
                                !!and to choose the sort order
                                !!It can be looked up, absence ⇒ 1
let personCodes = map ( Person with ( op = as samePerson, op < as
                                alphabetic ) → int ) { }

                                !!Also fast access paths for keys
type IndexRef is map ( string → Reference )
let masterIndex = map ( int → IndexRef ) {
    inits      → IndexRef { }
    fullInits → IndexRef { }
    names     → IndexRef { }
    fullNames → IndexRef { }

                                !!several procedures are required to
                                !!maintain these and use them
                                !!they are in a map to make extension easier
type IndexPack is structure (
    inserter: proc ( IndexRef, Reference )
    remover: proc ( IndexRef, Reference )
    finder: proc ( IndexRef, string → Reference )
    builder: proc ( Bibliography → IndexRef ) )
type MaintainsIndex is map ( int → IndexPack )
let maintainers = MaintainsIndex {
    inits → IndexPack (
        initInserter, initRemover, initFinder, initBuilder ) }

```

This example begins to show some of the structure that is suitable to represent the bibliographic database. To illustrate its use a sketch of parts of the procedures in the above structure is presented below. The reader should imagine that some input procedure *getReference*: **proc** (→ *Reference*) is used to obtain a new reference.

```

                                !!a field with this name is always present
let bibTypeFld = "bibtype"

                                !!The procedure to accept a new Reference
let acceptRef = proc ( ref: Reference )
begin
    if mainBib contains ref do knownAlready ( )
    let bibTy = ref ( bibTypeFld )' bibType

```

```

let reqs = requirements ( bibTy )
let defincd = dom1to1 [ string, BibField ] ( rcf )
if defincd contains reqs then
begin
                                !!validate it as a new reference
                                !!insert it in the main structures
    mainBib insert Bibliography { rcf }
    let author = rcf ( "authors" ) ' authors
    for each → aut in author do
    begin
        if not authorBibs contains aut do
            authorBibs insert AuthorBib { aut → Bibliography { } }
            authorBibs ( aut ) insert Bibliography { rcf }
        end
        let theKeys = rcf ( "keys" ) ' keys
        for each code → s in theKeys do
        begin
            let index = masterIndex ( code )
            if index contains s do
                s := extend ( s )
                index insert IndexRef { s — rcf }
            end
        end
    end
    else fieldsNotDefined ( rcf, reqs, bibTy, reqs -- defincd )
end

                                !!specific operations on indexes
let initInserter = proc ( i: IndexRef; rcf: Reference )
begin
    let theKeys = rcf ( "keys" ) ' keys
    if not theKeys contains inits do
    begin
        let iKey = makeInitsKey ( rcf )
        theKeys insert KeyMap { inits — iKey }
    end
    i insert IndexRef { theKeys ( inits ) — rcf }
end

let initRemover = proc ( i: IndexRef; rcf: Reference )
begin
    let theKeys = rcf ( "keys" ) ' keys
    if theKeys contains inits do
        i remove IndexRef { theKeys ( inits ) → rcf }
    end

let initFinder = proc ( i: IndexRef; s: string → Reference )
    if i contains s then i ( s )
    else notFound ( s )

let initBuilder = proc ( bib: Bibliography — IndexRef )
begin

```

```

let resIndex = IndexRef { }
for each ref in Bibliography do
  initInserter ( resIndex, ref )
resIndex
end

```

Though this illustrates the principal features of programming with maps, it also illustrates a problem. This problem can be seen from the strangely named function *dom1to1* which is defined below:

```

let dom1to1 = proc [ A, B ] ( m: map ( A → B ) → map ( A ) )
  map ( A ) { each a in m : a }

```

The problem is that we need to define others for maps with other degrees, e.g.

```

let dom1to2 = proc [ A, B, C ] ( m: map ( A → B, C ) → map ( A ) )
  map ( A ) { each a in m : a }
let dom1to3 = proc [ A, B, C, D ] ( m: map ( A → B, C, D ) → map ( A ) )
  map ( A ) { each a in m : a }
let dom2to1 = proc [ A, B, C ] ( m: map ( A, B → C ) → map ( A, B ) )
  map ( A ) { each a, b in m : a, b }
...

```

There appears to be no satisfactory notation at present⁴. Perhaps we should leave this to reflective techniques which generate the procedures when required.

⁴It may be possible to develop a variadic notation, for example $A^*;B^*$ could stand for as many A s that are supplied and as many B s. But this leaves notational difficulties within the program, see the body of the generator in *dom2to1*.

TOWARDS A UNIFICATION OF REWRITE BASED OPTIMIZATION TECHNIQUES FOR OBJECT-ORIENTED QUERIES

S. CLUET and C. DELOBEL

GIP Altaïr, B.P. 105, 78153 Le Chesnay Cedex, France

SUMMARY

This paper presents a formalism for the logical layer of an object-oriented query optimizer that subsumes two well known optimization approaches: the Orion technique based on classes extensions and the algebra based query rewritings. The formalism also allows easy and exhaustive factorization of common query subexpressions. Furthermore, it uses information on objects placement policies and indices to limit the search space for an equivalent expression, thereby reducing the rewriting phase.

1 INTRODUCTION

Current studies of object-oriented query optimization can be divided into three distinct approaches. The first stems from research on the Orion system [11] and has been extended in [12]. They consider the various class extensions involved in a query and evaluates what could be a simple selection in an object algebra through joins on these extensions. The second is based on algebraic query rewritings and has a large number of followers including [15], [4] and [16]. The last approach focuses on method code optimization [10].

In this paper, we propose a formalism that unifies the first two groups, optimization based on classes extensions and algebraic query rewritings.

The recipe is simple. It consists in introducing types in algebraic expressions and in reducing complex expressions representing selection, projection or joins criteria.

When these different techniques are combined, the number of possible expressions of a simple query is so large that it is fundamental to investigate heuristics to reduce the search space for an equivalent expression, thereby reducing the rewriting phase. The heuristics we advocate rely on the knowledge of indices [6] and objects placement policies [5]. These two kinds of informations can respectively be represented by paths of length equal or greater than one and by trees. Accordingly, we have chosen for algebraic expressions a DAG representation on which it is easy to map indices and placement trees.

The informations that we use to reduce query rewriting will be useful to the physical level of the optimizer in charge of finding the best implementation of a given algebraic expression and of evaluating its cost.

There exists a last technique of optimization that we have not mentioned and that has not been much considered yet: the factorization of common query subexpressions. This last technique is definitely important in an environment that involves complex structures and methods. There are two kinds of factorization. Local factorization consists in factorizing subexpressions common to one algebraic operation. Global factorization consists in factorizing subexpressions common to a sequence of algebraic operations that are liable to be evaluated by a unique algorithm (e.g. the nested loop algorithm that may evaluate a sequence of join, selection, projection operations). The formalism presented in [12] provides local factorization but does not consider global factorization.

We extended our formalism in a simple manner to provide global factorization. To each sequence of algebraic expressions that may be evaluated as a whole we associate a unique representation where the boundaries between subexpressions belonging to distinct algebraic operations are invisible.

Besides the formalization, our contribution is threefold. First, we expand algebraic expressions to introduce precise typing and intermediary variables. This results in a formalism that has the combined power of the algebraic and ORION approaches and provides local factorization. Also, we represent our algebraic expressions using DAG's which allows to highlight information on physical accesses such as indices or objects placement policies. Such information is crucial if one wants to limit the number of candidate equivalent query expressions to be considered. Last, we group some chosen sequences of algebraic operations in one unique representation. This provides the factorization of subexpressions common to distinct algebraic operators.

In the next section, we briefly describe the data model and languages that we will use through the paper. We explain our motivations and goals in Section 3. Section 4, 5 and 6 introduce our formalism in three stages. We conclude in Section 7.

2 PRELIMINARIES

We assume that the reader is reasonably familiar with object-oriented models and systems.

The Data Model

We consider here a model that is close to the O₂ data model [9]. The ideas apply of course to the optimization of query languages in other data models as well.

The model has classes and concrete types. Each class has a name, defines a structure and a behaviour. Classes obey the inheritance principle, they have an extension, i.e. the set of all their instances. Their instances are called objects and respect the encapsulation and identity principles. A concrete type only defines a structure. It may be named or not. Its instances are called values and know neither encapsulation nor identity. A concrete type does not have an extension. Concrete type can also be found in the Exodus data model [7].

The model supports inverse functions as found in Vbase [2].

Figure 1 presents a partial view of a database schema that we will use in the sequel. Below are some comments on the schema.

An object of class "Country" has two atomic attributes "name" and "continent" and an object attribute "capital". The class "Capital" is a subclass of "City". The attribute "country" has been redefined to be the inverse of attribute "capital" in class "Country". Set valued attributes are found in class "Informations". Names in lowercase represent concrete types while class names are capitalized. Class extensions are set values that are named after their class.

In this paper, we will consider nested indices as defined in [6]. The use that we make of these indices can be easily adapted to other types of indices.

Objects Placement Policies

The default objects storage consists of one single file where objects are stored in their order of creation. There is one physical record per complex value, string value or object. Each record has an address by which it can be referenced. Record corresponding to complex or string values can only be referenced once while those corresponding to objects may be referenced

Classes	Attributes
Country:	name: string, capital: Capital inverse of Capital::country, continent: string
City:	name: string, country: Country, brochure: Brochure, informations: Informations
Capital is a City:	country: Country inverse of Country::capital
Brochure:	map: Bitmap, text: Text
Informations:	hotels: Set(Hotel), restaurants: Set(Restaurant), day: activities, night: activities
Destination:	city: City, hotel: Hotel
....
Employee:	cv: CV, sales: set(Sale) inverse of Sale::employee,
CV:	name: name, birthplace: birthplace
Sale:	destination: Destination, number: integer, amount: float employee: Employee inverse of Employee::sales,

Figure 1: The Travel Agency Database Classes

several times. This is due to the identity principle respected by the latter and that allows sharing of objects.

The database administrator may define other storage policies. For instance, he may choose to have all the instances of class "Employee" stored together in one file. This kind of policy is the default in most object-oriented systems. The administrator may also want objects of one class to be stored along with one or more of their components. The following instruction notifies the system that objects of class "Brochure" must be stored in one single file and that each object record must be followed in the file by the records of its two components.

create group on Brochure: (map, text)

This last policy is the default one for values components of objects in the Exodus system. It is important to note that, due to the sharing of objects, it is not always possible for the system to follow exactly the placement policies.

In the O₂ system, placement policies are represented by trees.

Languages

Queries will always first be expressed in the O_2 query language. This language does not respect encapsulation in its *ad hoc* mode. An object may be considered as an object or as the value it encapsulates. In the first case, it will be queried through the methods it knows and in the latter by directly accessing its components.

The queries we will present do not use methods. This is why we have not defined any on the schema. We assume that a method invoked in a query has no side effects and thus that, as far as rewriting is concerned, a method call is comparable to a field selection. The difference between the two operations would lay in their arity but we will see that our formalism is able to treat n-ary as well as unary operations. The information on a method possible side effects can be obtained during method code compile time.

We will use algebraic operators that are similar to those defined in the ENCORE data model [15]. They are the following:

- $\text{Select}(A, \lambda a, p(a))$ selects the elements “a” of set “A” such that “p(a)” holds. Predicate “p” is a combination of the O_2 query language basic boolean functions.
- $\text{Project}(A, \lambda a, f(a))$ applies function “f” to the elements of set “A”. Function “f” is a combination of the O_2 query language basic functions (field selection, message sending, tuple construction, ...). This operator is a combination of the ENCORE “Image” and “Project” operators.
- $\text{Join}(A, B, \lambda a, b, p(a, b))$ joins sets “A” and “B” according to predicate “p”. If no predicate is given, the join operation is equivalent to a Cartesian product. In the relational algebra, the join operation takes two set of tuples and returns a set of tuples. Because of their data model, object algebras work differently. The join operation takes two sets of anything and returns a set of tuples. Thus, each join operation adds a level of tuple nesting. This unfortunately means the loss of the join associativity property as known in the relational model. It is a real handicap. In [8], we propose an *ad hoc* variation.

The operators are defined on and return set values in contrast with the ENCORE operators that are defined on and return set objects. Thus, the algebra we consider can be compared to complex objects algebras ([1], [14], [13]). The fact that we do not generate object identifiers eliminates the distinction that is made in the ENCORE algebra between identity and structural equivalences.

3 MOTIVATIONS AND GOALS

We intend to give a formal logical layer for an object-oriented query optimizer. We consider three distinct and interesting techniques and we propose a formalism that cover them all. The combination of these different techniques will make it possible to return a large number of expressions equivalent to a single query. Accordingly, we are also interested in finding heuristics to reduce the search space for an equivalent expression.

The first technique we consider, and that we will call the type based rewriting technique, has been developed for the Orion system [11] and extended in [12]. It works on the types involved in a query. In most object-oriented models, to a type corresponds an extension, i.e. the set of all its instances. The technique consists in performing joins between these different extensions to evaluate what could be a simple algebraic selection operation.

The second technique, which has also been used in relational systems, is the algebraic query rewriting based on equivalence of operators ([15], [4], [16]).

The last technique concerns common subexpressions factorization. A query in an object-oriented environment may invoke complex methods or follow long paths to access a given component. It would be a great gain not to evaluate the same method or follow the same path twice.

We first study these three approaches separately. Then, we summarize the desired features of our optimization model.

Three Different Approaches

In [11], P. Jenq, D. Woelk, W. Kim, W. Lee propose a graphical representation of Orion queries and tree traversal algorithms that allows rewritings based on types. This optimization concerns a distributed system but is also relevant in a centralized one. The queries are those of the Orion language first release [3] that can be translated by a selection operation in an object algebra. As a matter of fact, and this is its main drawback, the Orion technique does not concern other algebraic operations. This drawback has been partially overcome in [12]. Let us consider the following query:

Query Q1 What are the African capitals where one can scuba dive at day and swim at midnight?

```
select  c
from    c  in Capital
where   c.country.continent = "Africa" and
        "scuba diving" in c.informations.day and
        "midnight bathing" in c.informations.night)
```

□

Its corresponding Orion representation is given on Figure 2.

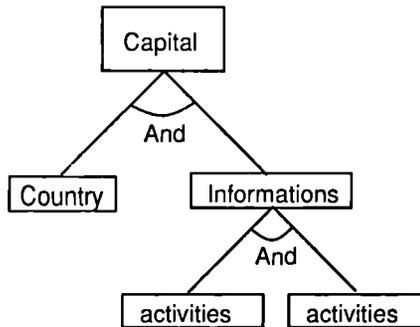


Figure 2: Orion representation of the query

The tree describes the complex types of the selection criteria. In the Orion data model, to a complex type corresponds an extension. Accordingly, to a node we may associate a set.

The tree is first reduced through decomposition into clusters. A cluster is a subtree whose root is an attribute node. The tree on Figure 2 may be reduced in two ways. We may consider the tree as a whole or build a cluster whose root is the "informations" node. There are three methods for evaluating a cluster.

- The forward traversal starts at the root node of a cluster. It consists of a projection on the children attributes, an evaluation of the qualified children that, then, allows

the qualification of their parent. This can be done algebraically (projection, selection, semi-join) or instance per instance.

- The backward traversal starts at the cluster leaves. The qualified attributes are evaluated through a selection. Then, a semi join returns the qualified parents.
- The mixed traversal combines the two previous methods. Some links are evaluated in forward traversal and others in backward traversals. For instance, in Figure 2 the query may be evaluated through a backward traversal from node “Country” to node “Capital” and then through a forward traversal toward node “Informations”.

These three kinds of traversals lead to a great number of possible evaluations of the simple Query Q1. It goes from a naïve instance per instance forward traversal to more complex evaluations involving joins. However, this approach lacks a formalism that would make it possible to draw a line between logical and physical optimizations.

We next consider the algebra based rewriting technique. In algebraic terms, the query is the following selection:

Query Expression E1

```
select (Capital, λ c (c.country.continent = “Africa” and
                    “scuba diving” in c.informations.day and
                    “midnight bathing” in c.informations.night))
```

□

On such a simple query, the only possible transformation is a partitioning of the selection operation. For instance, we can rewrite the query in the following manner:

Query Expression E2

```
Select (Select (Capital, λ c (c.country.continent = “Africa”))
        λ c2 (“scuba diving” in c2.informations.day and
             “midnight bathing” in c2.informations.night))
```

□

Assuming that there is an index on the path “country.continent” of the capitals, this rewriting would emphasize the use of that index.

However, there seems to be no way to generate, from this algebraic expression, the join operations that were detected by the Orion technique. Neither is it possible to emphasize the use one may make of the index we have on the path “continent” of the “Country” extension and the inverse property on the attribute “country” of class Capital. In other words, it seems that we cannot transform Expression E1 into the following expression, that first selects the African countries and work from then on to find the appropriate capitals:

Query Expression E3

```
Project (Select (Country, λ c (c.continent = “Africa”))
         λ c2 (“scuba diving” in c2.capital.informations.day and
              “midnight bathing” in c2.capital.informations.night)))
        λ c      c.capital)
```

□

This use of a partial index and of an inverse function is possible with the type based rewriting technique. It corresponds in Orion to a backward traversal from node “Country” to node “Capital”.

On the other hand, the algebraic approach is not limited to the rewriting of selection operations whereas the type based rewriting technique is. We will illustrate this with an example that figures a join operation that is transformed into a selection operation. This kind of transformation, although converse to the Orion ones, is not considered in the type based rewriting technique.

Let us consider the following example:

Query Q2 Find the employees who have sold travels to their city of birth.

```
select  e
from    e in Employee, s in e.sales
where  e.cv.birthplace.city = s.destination.city.name
□
```

The query algebraic translation is the following:

Query Expression E4

```
Project (Select (Join (Employee, Sale, λ e, λ s (s in e.sales))
                    λ t (t.e.cv.birthplace.city = t.s.destination.city.name))
        λ t (t.e))
□
```

Now, let us consider the two following equivalences supported by the Encore algebra.

Equivalence Eq1

Select (Join (A, B, λ a, b p(a,b)) λ t p_s(t.a,t.b)) ≡ Join (A, B, λ a, b (p(a,b) and p_s(a,b))) □

Equivalence Eq2

Project (Join(A, B, λ a, b (p(a, b))) λ t (t.a)) ≡ Select(A, λ a, ∃ b (b in Bs and p(a, b))) □

Equivalence Eq2 only applies if we do not consider duplicates in the resulting set. Using this two equivalences, we can eliminate the query join operation and consider instead the following simple selection:

Query Expression E5

Select(Employee, λ e, ∃ s (s in e.sales¹ and e.cv.birthplace.city = s.destination.city.name)) □

The gain brought by this transformation is that we do not have to consider all the sales of a given employee but, instead, we stop at the first sale that validates the predicate.

Before we go further, we would like to point out a problem one has to consider when studying query optimization in a object-oriented data model where some types do not have an extension. In Query Q2, the definition of Variable “s” depends on Variable “e”. In the corresponding join operation, Variable “s” is defined on its membership in the “Sale” extension and the variables dependence is translated with the predicate “(s in e.sales)”. Now, supposing that the type associated with variable “s” did not have an extension, this would appear to forbid an algebraic translation. A solution to this problem is to consider two kinds of extensions. Extensions of the first kind are maintained by the database system while extension of the second are not but could be evaluated at a huge cost by scanning the database. We call “virtual extensions” the extensions belonging to the second category. In the data model we are considering, concrete types extensions are virtual extensions. The rules that manage these

¹Since “s in e.sales” is more specific, “s in Sale” has been omitted

unordinary sets are simple enough. Select, projection, union and difference operations are not defined on virtual sets. Join operations between two virtual sets are not considered. One virtual set is accepted in a join operation if the join condition contains a membership test on the variable associated to the virtual set. For instance, the following join expression is acceptable:

Query Expression E6

Join (Informations, activities, $\lambda i, a, (a \text{ in } i.\text{day})$) \square

The type based rewriting technique combined to the algebraic formalism will make it possible to return a large number of expressions equivalent to a single query. In the environment we are considering, the knowledge of placement policies makes it possible to eliminate some of them as not relevant. For instance, let us consider again the query represented on Figure 2 and let us suppose that the database administrator has specified the following storage policy:

create group on Capital: (Informations (day, night))

This means that all objects of class “Capital” should be stored along with their “Informations” component and followed by this component attribute values. In that case, and in the absence of appropriate indices, we know that it will be more appropriate to evaluate the right part of the tree instance per instance in a forward manner rather than performing a join between the “capital” and “Informations” extension. For the same reasons, to partition the two selection conditions concerning the informations attribute should not be considered.

Now, we will study the third technique, the factorization of common query subexpressions.

Let us consider the following query:

Query Q3 Find the employees who have sold travels to their city of birth. Give their names and cities of birth. The travels must be special offers or have prices greater or equal to \$20,000.

```
select  tuple(name: e.cv.name, city: e.cv.birthplace.city)
from    e in Employee, s in e.sales
where   e.cv.birthplace.city = s.destination.city.name and
        (s.destination = special_offer or s.amount >= 20000)
```

\square

This query can be translated into the following algebraic expression:

Query Expression E7

```
Project (Select (Join (Employee, Sale,  $\lambda e, s (s \text{ in } e.\text{sales})$ )
                 $\lambda t (t.e.cv.birthplace.city = t.s.destination.city.name and$ 
                 $(t.s.destination = special\_offer or t.s.amount >= 20000)))$ 
         $\lambda t (tuple(name: t.e.cv.name, city: t.e.cv.birthplace.city)))$ )
```

\square

Let us now emphasize anomalies that the two approaches we have studied so far are powerless to solve. The operation “e.cv.birthplace.city” is performed twice on the qualified employees, once to test a selection condition and the second time to perform the projection operation. Another point is that this operation is evaluated for every pair (employee, sale) while it only concerns employees. One must remember that in an object oriented environment, a simple field selection may cause a page fault. Thus, it might be a gain to factorize all possible operations especially if this can be done at a lesser cost. However, it seems that none of the two approaches we previously studied can provide an appropriate treatment of these anomalies.

We plan to overcome this drawback.

To Summarize

We are considering a formalism for the logical layer of an optimizer. We want it (i) to subsume the type based rewriting technique and the algebraic approach, (ii) to support informations on objects placement policies and indices in order to reduce the rewriting phase and (iii) to allow easy and exhaustive factorization of common subexpressions and to emphasize subexpressions depending of one variable in a multi-variables query. We will present our formalism in three stages, each one corresponding to one of our three goals.

4 A SIMPLE IDEA: A TYPED ALGEBRA

To each possible traversal of the Orion tree representing a query, one may associate an algebraic expression. If we consider the query represented on Figure 2, a forward instance per instance traversal corresponds to Expression E1, a mixed traversal starting from node "Country" going backward to node "Capital" and then forward, instance per instance, from node "Capital" to the nodes "activities" may correspond to Expression E3.

However, we have seen that, using algebraic equivalences, it seemed impossible to go from the first algebraic expression to the second. Neither was it possible to generate joins operations from a simple selection. These limitations are due to the lack of a good typing information.

To overcome this shortcoming, a simple idea is to consider all the elementary operations involved in an algebraic expression and type them.

We illustrate this on Expression E1 that is appropriately expanded to incorporate the necessary typing:

Query Expression E8

```
Select (Capital,  $\lambda$   $c_a$ ,  $\exists$   $c_{ou}$ ,  $c_{ont}$ ,  $i$ ,  $a_d$ ,  $a_n$ ,
      ( $c_{ou}$  in Country and  $c_{ont}$  in string and  $i$  in Informations and
        $a_d$  in activities and  $a_n$  in activities and
        $c_{ou} = c_a.country$  and  $c_{ont} = c_{ou}.continent$  and  $i = c_a.informations$  and
        $a_d = i.day$  and  $a_n = i.night$  and
        $c_{ont} = "Africa"$  and "scuba diving" in  $a_d$  and "midnight bathing" in  $a_n$ ))
```

□

On the first line, we associated a variable to each elementary operation. On the second and third lines, we specified the variables types by their membership in virtual or real extensions. The next two lines define the links between variables and the last line expresses the initial selection conditions.

In more formal terms, the selection operation has been expanded in the following manner:

$$\text{Select}(A, \lambda a, \exists v_1, v_2, \dots, v_n \\ (\text{p}_{type}(v_1, v_2, \dots, v_n) \text{ and } \text{p}_{def}(a, v_1, v_2, \dots, v_n) \text{ and } \text{p}_{sel}(a, v_1, v_2, \dots, v_n)))$$

Predicate p_{type} specifies the variables types and Predicate p_{def} defines the links between the different variables. It is important to note that to define these links is to express all the elementary operations one has to perform. Predicate p_{sel} is the initial selection condition.

We note that every variable represents a different operation. The expression "c.informations" that figured twice in the original selection has been factorized and appears only once in this new translation. However, this factorization does not give us entire satisfaction since it is just local to an operation. We will come back later to this problem.

Now, let us consider the new algebraic equivalence Eq3.

Equivalence Eq3

$\text{Select}(S1, \lambda s \text{ s.a in } S2) \equiv \text{Project}(S2, \lambda a \text{ inverse}(a)) \quad \square$

If we apply this equivalence to Expression E8, we may generate a typed version of Expression E3. We previously used equivalence Eq2 to transform a join operation into a selection. It can also be used the other way and allows us to generate as many joins as possible between the various extensions. For instance, we may generate Expression E9.

Query Expression E9

$$S_{\text{join}} \leftarrow (\text{Join}(\text{Capital, Informations, } \lambda c_a, i, \exists c_{ou}, c_{ont}, a_d, a_n, \\ (c_{ou} \text{ in Country and } c_{ont} \text{ in string and } a_d \text{ in activities and} \\ a_n \text{ in activities} \\ c_{ou} = c_a.\text{country and } c_{ont} = c_{ou}.\text{continent and } i = c_a.\text{informations} \\ \text{and } a_d = i.\text{day and } a_n = i.\text{night and} \\ c_{ont} = \text{"Africa" and "scuba diving" in } a_d \text{ and} \\ \text{"midnight bathing" in } a_n))$$

$$\text{Result} \leftarrow \text{Project}(S_{\text{join}}, \lambda t, \exists c_a, \\ (c_a \text{ in Capital and} \\ c_a = t.c_a), \\ c_a))$$

\square

This expression consists of a join operation, whose condition is the initial selection condition, and of a projection. Other equivalences can be applied to this expression to introduce new selection or joins operations.

We notice that the typing we performed on the selection operation have also been performed on the join and projection operations. The Encore algebraic equivalences are adapted to this new formalism in a straightforward manner. Details on this formalism can be found in [8].

At this stage we have reached our first goal. By introducing typing and intermediary variables in the algebraic expressions, we made it possible to use equivalences that could not be applied otherwise (e.g. equivalences Eq2 and Eq3 on expression E1) and that generate expressions equivalent to the Orion tree traversals. It is obvious that all the other algebraic equivalences are still applicable. Accordingly, we have a formalism that subsumes the Orion technique and the traditional algebraic formalism.

Another major goal was to support informations on object placement policies and indices in order to reduce the rewriting phase. Since these informations are represented by trees and paths, we chose a DAG representation for the algebra.

5 GRAPHICAL REPRESENTATION OF A TYPED ALGEBRA

Let us study the graphical representation of Expression E8 given on Figure 3. For the moment, we will ignore the dashed lines represented on this figure. The selection operation consists of (i) a graph figuring the selection variables, their types, the way they are linked and (ii) a tree representing the selection condition.

The graph has two different kinds of node. The oval node represents the set on which the selection is made and its associated variable (here c_a : Capital). Each rectangular node

represents an intermediary variable (c_{ou} , c_{ont} , i , a_d , a_n) and its type (p_{type} in the textual representation).

The edges represent the operations that link the variables (p_{def} in the textual representation). There cannot be two edges representing the same operation starting from the same node. This guaranties an exhaustive local factorization and at a lesser cost since the number of edges starting from one node is limited by the type of the node. The operations of the example are unary. However, as we will see later, we can also represent n-ary operations (method calls, tuple construction, ...). An n-ary operation is represented by “n” numbered edges having same destination.

The initial selection predicate (p_{sel}) is represented by a tree whose leaves are the variables or the constants on which the predicate is defined.

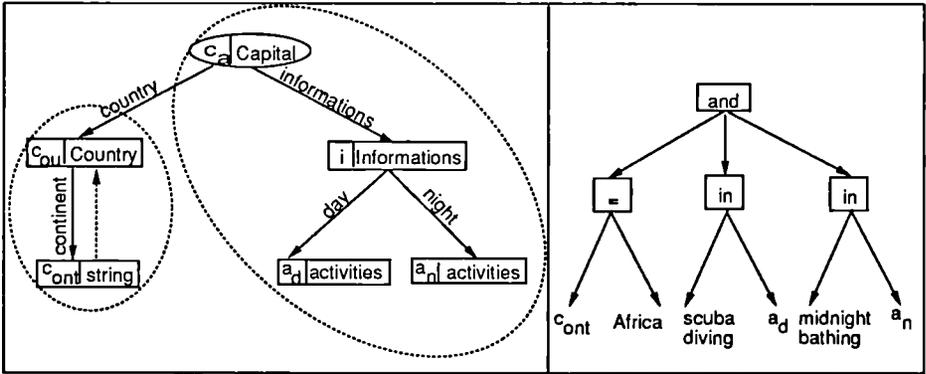


Figure 3: An Adorned Graphical Representation of Expression E8

Now, let us consider the dashed lines that represent system informations. On the figure, areas bordered by a dashed line indicate placement trees and the hatched edge represents an index. These informations, as we will illustrate it now, are used to limit the rewriting phase.

The first rule to limit rewriting is given below.

Rule R1

In the absence of appropriate indices, it is preferable not to introduce joins on nodes that are in a same hatched area.

□

For instance, Nodes c_a , i , a_d and a_n are in a same area that does not contain any index edge. Thus, it is better not to introduce a join between the “Capital” extension of Node c_a and the “Informations” extension of Node i . Accordingly, Expression E9 cannot be generated from this DAG representation. This restriction can easily be understood by the fact that, according to the placement algorithm, the values corresponding to the nodes figuring in a same area are stored in sequential order in the same file.

Another rule concerning placement policies is the following:

Rule R2

In the absence of appropriate indices, it is better not to partition a selection in two if the

corresponding node are in a same hatched area.

□

For instance, we will not allow a partitioning of the selection operation that would cut the tree formed by Nodes i , a_d and a_n into two.

If we do not consider the ordering of the conditions expressed in a selection, rules R1 and R2 leave us with four possible expressions of the query. The one represented on the graph, Expression E3, the following one that we do not type for clarity, and another one that is pretty much the same except for the use of an inverse function:

Query Expression E10

```
Project (Join (Select (Country,  $\lambda c_{ou}, c_{ou}.continent = \text{"Africa"}),$ 
                Select (Capital,  $\lambda c_a, \text{"scuba diving" in } c_a.informations.day \text{ and}$ 
                         $\text{"midnight bathing" in } c_a.informations.night)$ 
                 $\lambda c_a, c_{ou}, c_a.country = c_{ou}$ )
         $\lambda t t.c_a)$ 
```

□

We have seen some ways to use placement policies to reduce rewriting. This can also be done by simply using index informations. For lack of place, we will not illustrate this possibility but details can be found in [8].

We now summarize what has been achieved. User queries are transformed into typed algebraic operations that are represented graphically and that support informations on object placement policies and indices. The transformations we perform are based on algebraic equivalences and limited by rules concerning object placement policies and indices. In other words, our optimizer rewriting rules are of the following form:

algebraic condition, system condition \rightarrow algebraic transformation

The algebraic condition specifies the structure of the expression before its transformation, the system condition prohibits transformation when the objects placement policy and indices context is not right and the algebraic transformation specifies the resulting algebraic expression.

At this stage we have reached two out of three goals. We will now see how to achieve global factorization.

6 A GLOBAL REPRESENTATION FOR A GLOBAL FACTORIZATION

Sequences of join, selection and projection operations are liable to be evaluated by a same algorithm. For instance, Expression E7 can be evaluated by a single nested loop algorithm. In these sequences, one may find subexpressions common to distinct algebraic operation (e.g. "e.cv.birthplace.city" common to the selection and projection operations of Expression E7). The type based and algebra based rewriting techniques are powerless to detect and factorize these expressions because they consider separately each algebraic operation.

To solve this problem, a simple idea consists in having a global representation for each sequence of algebraic expressions that may be evaluated as a whole. In this global representation, the boundaries between subexpressions belonging to distinct algebraic operations are invisible. This allows exhaustive factorization whereas the previous approaches only considered local factorization.

Currently, we only consider sequences of join-selection-projection operations.

We will illustrate our meaning by considering the global representation of Expression E7 given on figure 4.

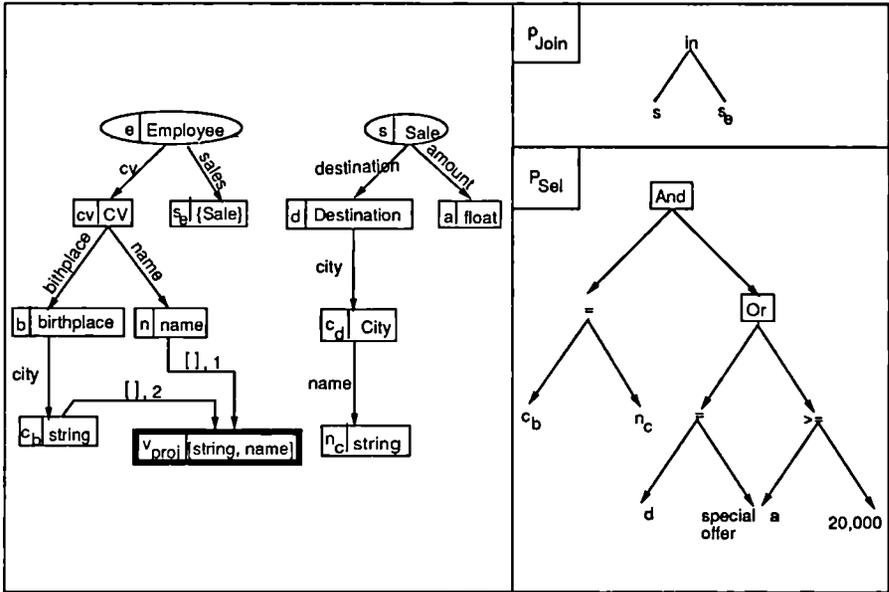


Figure 4: A graphical representation of Expression E7

For clarity, we have not adorned the representation. It figures the three algebraic operations of join, selection and projection. It consists of (i) two subgraphs whose roots represent the sets on which the join operation is performed and whose other nodes figure intermediary variables representing elementary operations, their types, the way they are linked, (ii) a tree representing the join condition and (iii) a tree representing the selection condition. The variable representing the projection operation is in a bold rectangle (v_{proj}). In this example, the two subgraphs rooted at Nodes “Employee” and “Sale” are disjoint. However, this is not always the case.

The graph has two different kinds of nodes. The oval nodes represent the sets on which the operations are defined and their associated variables (e, s) and the rectangular nodes represent the intermediary variables and their types ($cv, b, c_b, n, v_{proj}, s_e, d, c_d, n_c, a$). The edges represent the operations that link the variables. One may notice the representation of an n -ary operation. The function that constructs a tuple having two attributes is represented by two labelled edges toward variable v_{proj} .

This grouped representation of three sequential algebraic operations allows an exhaustive factorization of the query common subexpressions that would not have been possible by considering each operation separately. For instance, the subexpression “ $e.cv.birthplace.city$ ”, that is represented by the intermediary variable c_b , is common to the selection and projection operations and has been factorized. One can also notice two disjoint subgraphs. The first that includes Nodes e, cv, b, c_b, n and v_{proj} concerns the operations on the “Employee” objects. The second, that includes Nodes s, d, c_d, n_c and a concerns the operations on the “Sale” objects. These disjoint subgraphs indicate that these operations do not have to be evaluated for every pair (employee, sale) resulting of the join operation and that it would be a gain to push them out of the join operation.

It is important to understand that this factorization serves as an indication to the physical layer of the optimizer that may ignore it to implement an algorithm that better serves its purpose.

So, once again, let us summarize what has been achieved.

User queries are transformed into typed algebraic expressions that are represented graphically. The typed algebra is now limited to three operators: union, intersection and an operator representing three ordered and sequential operations of join-selection-projection that we call JSP.

The textual expression of this operator is as follows:

$$\text{JSP}(A, B, \lambda a, b, \exists v_{proj}, v_1, v_2, \dots, v_n \\ (\text{p}_{type}(v_{proj}, v_1, v_2, \dots, v_n) \text{ and } \text{p}_{def}(a, b, v_{proj}, v_1, v_2, \dots, v_n) \text{ and} \\ \text{p}_{join}(a, b, v_{proj}, v_1, v_2, \dots, v_n))) \text{ and} \\ \text{p}_{sel}(a, b, v_{proj}, v_1, v_2, \dots, v_n), \\ v_{proj}))$$

Variables a and b are defined on the two sets we are considering. Variables $v_{proj}, v_1, v_2, \dots, v_n$ are intermediary variables representing the join, selection and projection elementary operations. Predicate p_{type} defines the intermediary variables types. Predicate p_{join} is the join predicate and predicate p_{sel} is the selection predicate. Variable v_{proj} represents the projection operation.

It can be noted that this operator is also able to represent single selection ($B = \emptyset$, $\text{p}_{join} = \text{true}$ and $v_{proj} = \epsilon$), projection ($B = \emptyset$, $\text{p}_{join} = \text{true}$ and $\text{p}_{sel} = \text{true}$), and join ($\text{p}_{sel} = \text{true}$ and $v_{proj} = \epsilon$) or any of the following combinations: join-selection, join-projection and selection-projection.

The transformation from a select-from-where user query to a graphical typed algebra expression is straightforward. Once this transformation is made we have succeeded in a global factorization that we will keep through the rewriting process.

As we specified it, rewriting rules comport three parts. Two that are traditional (algebraic condition and algebraic transformation) plus one that concerns system informations and that will be used to reduce the rewriting phase. The Encore algebraic equivalences are easily adapted to the new formalism as shown in [8].

As we have seen, the intermediary variables we included in the algebraic expressions and the appropriate type informations give us the combined power of algebra based and type based rewriting techniques.

So, finally, we have reached the three goals we defined in section 3.

7 CONCLUSION

In this paper, we have presented a formalism that covers the rewriting phase of a query optimizer for an object-oriented database system. It consists of a DAG typed algebra that subsumes the Orion optimization technique [11] and the traditional algebraic formalism [15], [4] and [16].

This formalism supports informations on objects placement policies and indices that are used to considerably reduce the rewriting phase. These informations are also useful to the physical level of the optimizer in charge of finding the best implementation of a given algebraic expression and of evaluating its cost.

The formalism we proposed also allows a simple and exhaustive factorization of the common subexpressions of a query and emphasizes the expressions that only depends of one variable in a multi-variables query.

References

- [1] S. Abiteboul and C. Beeri. *On the Power of Languages for the Manipulation of Complex Objects*. Technical Report, INRIA and the department of computer science of the Hebrew University of Israel, 1987.
- [2] T. Andrews and C. Harris. Combining Language and Database Advances in an Object-Oriented Development Environment. In *Proc. OOPSLA, Orlando, Florida*, October 1987.
- [3] J. Banerjee, W. Kim, and K. Kim. *Queries in Object-Oriented Databases*. Technical Report DB 188-87, MCC, Austin, Texas, June 1987.
- [4] C. Beeri and Y. Kornatzky. Algebraic Optimization of Object-Oriented Query Languages. In *Proc. ICDT, Paris, France*, 1990.
- [5] V. Benzaken and C. Delobel. Enhancing Performance in a Persistent Object Store: Clustering Strategies in O₂. In *Proc. POMS, Martha's Vineyard, Massachusetts, USA*, September 1990.
- [6] E. Bertino and W. Kim. Indexing Techniques for Queries on Nested Objects. *IEEE Transaction Knowledge and Date Engineering*, January 1989.
- [7] M. Carey, D. DeWitt, and S. Vandenberg. A Data Model and Query Language for EXODUS. In *Proc. SIGMOD, Chicago, Illinois*, 1988.
- [8] S. Cluet. *Langages et Optimisation de requêtes pour Systèmes de Gestion de Base de données orienté-objet*. PhD thesis, Université de Paris-Sud, 1991. to appear.
- [9] O. Deux et al. The Story of O₂. *IEEE Transaction on Knowledge and Date Engineering*, 2(1), March 1990.
- [10] G. Graefe and D. Maier. *Query Optimization in Object-Oriented Database Management Systems with Encapsulated Behaviour*. Technical Report, Oregon Graduate Center, 1989.
- [11] P. Jenq, D. Woelk, W. Kim, and W. Lee. Query Processing in Distributed ORION. In *proc. EDBT, Venice, Italy*, March 1990.
- [12] A. Kemper and G. Moerkotte. Advanced Query Processing in Object Bases Using Access Support Relations. In *proc. VLDB, Brisbane, Australy*, 1990.
- [13] G. Kuper and M. Vardi. A New Approach to Database Logic. In *proc. 3rd ACM PODS*, 1984.
- [14] H. J. Schek and M. H. Scholl. The Relational Model with Relation-Valued Attributes. *Information Systems*, 11(2), 1986.
- [15] G. Shaw and S. Zdonik. An Object-Oriented Query Algebra. In *Proc. DBPL, Salishan Lodge, Oregon*, June 1989.
- [16] D. Straube and T. Özsu. *Queries and Query Processing in Object-Oriented Database Systems*. Technical Report, Department of computing science, university of Alberta, Edmonton, Alberta, Canada, 1990.

OBSERVATION OF PHOTOLUMINESCENCE FROM InAs SURFACE QUANTUM WELLS GROWN ON InP(100) BY MOLECULAR BEAM EPITAXY

Z.Sobiesierski, S.A.Clark, R.H.Williams
Department of Physics
University of Wales College of Cardiff
P.O.Box 913
Cardiff CF1 3TH, Wales, U.K
Tel. + 44 222 874000

A.Tabata, T.Benyattou, G.Guillot
LPM (URA CNRS 358),
INSA de Lyon
69621 Villeurbanne Cedex, France.
Tel. + 33 72 43 81 61

M.Gendry, G.Hollinger, P.Viktorovitch
LEAME (URA CNRS 848)
Ecole Centrale de Lyon
BP 163-69131
Ecully Cedex, France.
Tel. + 33 78 33 81 27

SUMMARY

Photoluminescence (PL) measurements are presented for thin epitaxial layers of InAs, $2.5\text{\AA} < d < 36\text{\AA}$, grown on InP(100) by molecular beam epitaxy (MBE). The combination of efficient carrier capture and PL red-shift with increasing InAs thickness clearly indicate the formation of InAs quantum wells on the InP surface. Data are also presented for InAs/InP structures capped with strained layers of either GaAs or $\text{In}_{0.5}\text{Al}_{0.5}\text{As}$ and for InAs/InP layers subjected to oxidation with time. Since radiative recombination within the InAs layers can be distinguished from PL arising from both bulk and surface defects, this system allows us to monitor the quality of both the InAs/InP and InAs/air interfaces via their influence on the InAs quantum well luminescence.

INTRODUCTION

The physical properties of surfaces are normally explored by using experimental techniques which have an inherent sensitivity to the first few atomic layers of the material. In this case, the absolute strength of the signal which we measure depends mainly on the relatively short distance over which the probe beam is absorbed, or through which the excitation can propagate without extinction. However, more bulk-sensitive optical techniques such as photoluminescence, photocurrent, or even Raman scattering measurements become probes of the local electronic environment in semiconductor quantum well structures. In particular, the intensity of photoluminescence (PL) measurements depends not only on the absorption coefficient at the excitation energy, but also the distance over which carriers diffuse before becoming confined in their respective potential wells. The possibility of using a quantum well at a semiconductor surface as an efficient light emitter has already been demonstrated [1], by growing InP on graded $\text{Ga}_x\text{In}_{1-x}\text{P}$. These experiments found that non-radiative surface recombination appeared to be saturated by the high density of carriers collected in the well, and so played a minor role. Additionally, the quantised energy levels in a potential well placed at, or near, a semiconductor surface can couple to electronic states at the

surface itself. For example, the interaction of surface and near-surface GaAs/Ga_{0.7}Al_{0.3}As quantum wells with the free surface has revealed coupling of the confined states with surface states located near the band edges [2]. Hence, in principle, it is possible to probe the physical extent [3] of localised wavefunctions. In this paper we consider the optical properties of thin, strained InAs quantum wells formed on InP surfaces. Pseudomorphic InAs/InP surface wells can be formed as a result of As/P exchange [4], when InP wafers are heated under an As₄ over-pressure. However, far superior control over the InAs/InP interface can be achieved if the InAs layers are deposited by molecular beam epitaxy. Since the InAs-related PL transition energies will depend on the InAs well width, we can readily distinguish radiative recombination within the InAs layers from deep level luminescence in the InP substrate material.

EXPERIMENTAL PROCEDURES

Growth of InAs was carried out in two separate MBE reactors, a VG Semicon V80H and a Riber 2300, using slightly different conditions. Good overall agreement was found between similar structures grown in either machine. Since the results to be presented in this letter pertain, on the whole, to material grown in the VG Semicon machine, we will restrict our discussion of growth details to this system [5]. Fe-doped, semi-insulating InP(100) substrates were obtained from Nippon Mining and prepared by etching in a mixture of H₂SO₄:H₂O₂:H₂O in the ratio 7:1:1. Prior to growth, the substrate temperature was slowly increased from ambient, using a stabilising As₄ over-pressure of 8 x 10⁻⁵ mbar, whilst the InP surface was observed in-situ by Reflection High Energy Electron Diffraction (RHEED) measurements. At low temperature the RHEED pattern exhibited only bulk streaks; however, at 500^o C, an abrupt transition was seen to a (2x1) surface reconstruction indicative of oxide removal from the InP. The substrate temperature was then increased to 510^o C and epitaxial growth of InAs initiated using a growth rate of 0.52 m/hr (previously calibrated using a combination of double crystal x-ray diffraction measurements and scanning electron microscopy) and a V:III pressure ratio of 120:1. As noted earlier, oxide removal under an As over-pressure results in the creation of a thin InAs layer, ~5Å thick, on the InP surface. The thickness of this InAs layer, formed by the exchange of P for As, obviously needs to be added to the deposited thickness of epitaxial InAs to give the total InAs thickness. Throughout the text, we will only refer to the thickness of InAs deposited epitaxially by MBE when distinguishing between samples. The total InAs thickness will only be stated explicitly when we seek to identify a trend for the whole range of samples grown, e.g. the variation of PL peak energy versus InAs layer thickness. PL measurements were performed in the temperature range 4.2K to 200K inside a cryostat, using an excitation density of ~50 mW/cm² at 632.8nm. A Ge detector was used for PL energies greater than 0.7 eV, whilst a PbS cell was used for PL energies below 0.7 eV. The spectral resolution of the system always remained better than 2 meV.

RESULTS AND DISCUSSION

Figure 1 compares 4.2K PL spectra for 10Å, 5Å and 2.5Å thick layers of InAs deposited on InP(100), with the PL spectrum obtained from an InP substrate chemically etched with choline. These deposited InAs thicknesses were checked by ex-situ x-ray photoelectron spectroscopy (XPS). The form of the XPS attenuation plots for the P core levels indicated average layer-by-layer growth of the InAs on InP, for these thicknesses. All the spectra for InAs/InP in figure 1 exhibit more PL from the InAs layer than from the

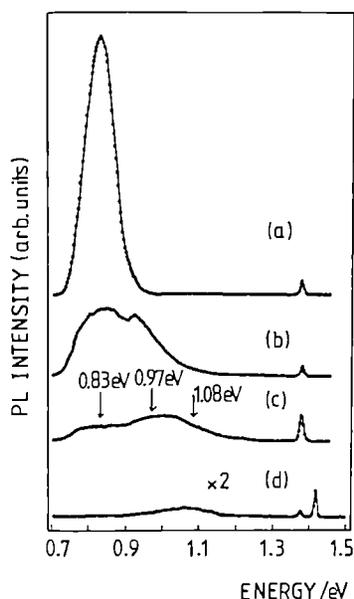


Figure 1: Comparison of 4.2K PL spectra for (a) 10Å InAs/InP, (b) 5Å InAs/InP, (c) 2.5Å InAs/InP and (d) InP etched in choline.

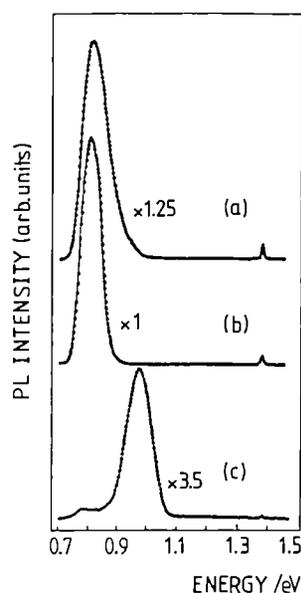


Figure 2: 4.2K PL spectra for (a) 10Å InAs/InP, (b) 5Å GaAs/10Å InAs/InP and (c) 200Å In_{0.5}Al_{0.5}As/6Å InAs/InP, showing effect of capping layer.

InP substrate, even though the absorption length at this excitation energy is around 170 nm for InP [6]. PL spectra for 20Å and 30Å layers of InAs/InP show a continuous red-shift of the PL emission to lower photon energies, together with an accompanying decrease in PL linewidth, with increasing layer thickness. The combination of efficient carrier capture and PL red-shift with increasing InAs thickness clearly indicate the formation of InAs quantum wells on the InP surface. Whilst the PL lineshape for the 10Å InAs/InP surface quantum well can be adequately described by a single Gaussian, the PL lineshapes observed for the thinner InAs layers are similar to those previously attributed to monolayer fluctuations in ultra-thin quantum well structures [7]. Since the bulk lattice constants of InP, InAs are $a_0 = 5.8687 \text{ \AA}$ and $a_0 = 6.0583 \text{ \AA}$ respectively, the 2.5Å, 5Å and 10Å epitaxial InAs layers differ in thickness by ~ 1 ML. We have found that all the InAs associated luminescence in figure 1 can be fitted with weighted contributions from three simple Gaussians located at energies of $\sim 0.83 \text{ eV}$, $\sim 0.97 \text{ eV}$ and $\sim 1.08 \text{ eV}$. The full-width-half-maximum (FWHM) value of the PL lineshape obtained for 10Å InAs/InP is $93 \pm 1 \text{ meV}$, which can be compared with a linewidth of 10-14 meV obtained for 1-3 ML InAs/InP quantum wells grown by metal-organic-vapour-phase-epitaxy (MOVPE) [8,9,10]. For InAs quantum wells of this thickness, the PL FWHM will be influenced by the material quality of the well and the barriers, as well as by interfacial roughness [11]. Neither the InAs/InP nor InAs/air interfaces provide abrupt potential barriers. The smoothness of the InAs/InP interface is controlled by the As/P exchange reaction, which itself depends on the substrate preparation and exact conditions of oxide removal prior to growth. The nature of the InAs/air interface is determined by the oxidised InAs surface. We have attempted to examine the effect which this oxide layer has on the quantum well emission by depositing strained "capping" layers of either GaAs or In_{0.5}Al_{0.5}As directly after stopping InAs growth, thus moving the oxide layer further away from the outermost InAs interface.

Figure 2 compares the 4.2K PL spectrum for 10Å InAs/InP with PL spectra obtained from "capped" surface quantum wells, 5Å GaAs/ 10Å InAs/InP and 200Å In Al As/6Å InAs/InP. In all three cases, the structures were still observed to be strained after growth. The main thing to note for the GaAs-capped layer is that the PL FWHM has decreased from 93 ± 1 meV for 10Å InAs/InP to 63 ± 1 meV for 5Å GaAs/10Å InAs/InP, whilst there is little change in peak position. The PL emission for 200Å In Al_{0.5}As/6Å_{0.5}InAs/InP is centred at ~ 0.97 eV, with a FWHM of 98 ± 1 meV. In this case, the broad lineshape reflects the sensitivity of the InAs quantum well emission to alloy broadening in the capping layer. Furthermore, the lack of any appreciable shift in PL energy between an InAs surface quantum well and a similar structure capped with either GaAs or InAlAs suggests that any coupling between the confined states in the InAs quantum well and the InAs surface states is not a significant factor [12].

Figure 3 plots the observed PL energy (left hand axis) together with the integrated InAs-to-InP PL intensity ratio (right hand axis) versus total InAs thickness (epitaxial InAs + 5Å InAs due to As/P exchange). The solid curve has been calculated assuming the PL transition to be $e_1 - hh_1$ within strained InAs quantum wells with InP barriers. The parameters used in the calculation are listed in Table 1. The value of conduction band offset chosen, $\Delta E_c = 0.4$, is that recently used to model InAs quantum wells with InP barriers grown by MOVPE [10]. The theoretical calculation provides quite a good representation of the experimental data, particularly as we have: (i) assumed that the potential wells are exactly rectangular, (ii) not included any strain relaxation for the thicker wells and (iii) not corrected for the exciton binding energy. The integrated PL intensity in figure 3 is observed to reach a maximum value for a total InAs thickness of 15Å and thereafter decreases with increasing InAs thickness. This decrease in PL intensity is almost certainly a sign of strain relaxation via the generation of dislocations. In-situ RHEED measurements of the InAs surface lattice constant have confirmed the onset of relaxation for InAs thicknesses around 17Å. Our results are in good agreement with PL measurements for InAs/InP single quantum wells grown by MOVPE, where evidence for strain relaxation has been found for well thicknesses greater than 5ML InAs [9].

Parameter	InAs	InP
a_0 (Å)	6.0583	5.8687
C_{11} (10^{10} Pa)	8.33	10.22
C_{12} (10^{10} Pa)	4.53	5.76
a (eV)	-6	-6.4
b (eV)	-1.8	-2
E_g (eV)	0.472 (strained)	1.423
m_e (m_0)	0.0325 (strained)	0.085
m_{hh} (m_0)	0.35 (strained)	0.45

TABLE 1 Parameters used to calculate $e_1 - hh_1$ transition energy for strained InAs/InP quantum wells, taking $\Delta E_c = 0.4$.

We have also measured the temperature dependence of the InAs-related luminescence signal. The integrated intensities of these PL bands are plotted in Figure 4, as a function

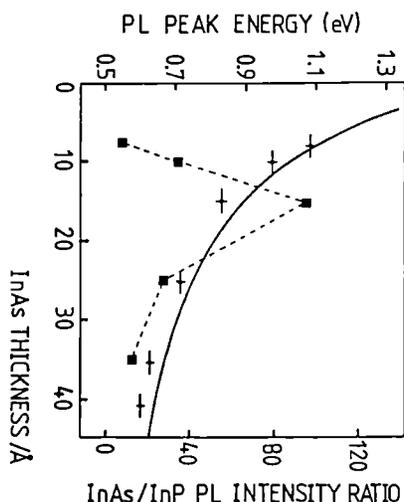


Figure 3: Variation of PL peak energy (+) and InAs-to-InP PL intensity ratio (■) with total InAs layer thickness. Solid line is calculation described in text.

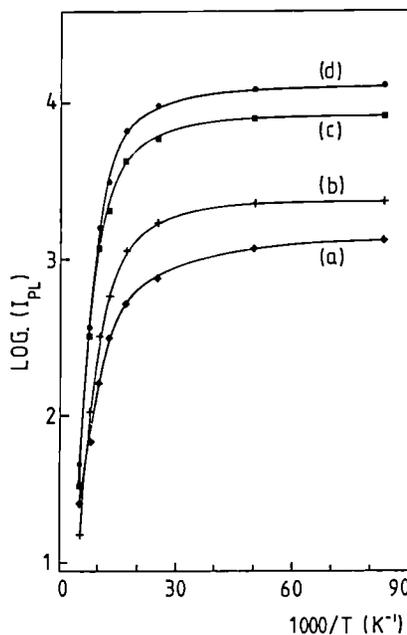


Figure 4: Temperature dependence of the integrated quantum well luminescence signal for (a) 2.5 Å InAs/InP, (b) 5 Å InAs/InP, (c) 7.5 Å InAs/InP and (d) 10 Å InAs/InP.

of temperature, for samples consisting of 2.5 Å InAs/InP, 5 Å InAs/InP, 7.5 Å InAs/InP and 10 Å InAs/InP. The shape of such Arrhenius plots of $\log(I_{PL})$ against inverse temperature, together with the changes in shape as related to well width, can be used to infer the main mechanisms responsible for non-radiative recombination. For example, a linear slope at high temperatures, which changes with well width, suggests that thermal activation of the carriers out of the quantum well is followed by non-radiative recombination within the barrier or substrate material [13]. However, such analysis can only be applied if the quality and definition of the InAs quantum wells are sufficient to provide temperature-independent, non-radiative losses within the wells themselves. In our case, the gradient of the curves at low values of $1000/T$ increases, as expected, with increasing well width, but the activation energies deduced from these slopes fall significantly below the values calculated from the PL transition energies we observe. A large part of this discrepancy arises because we are unable to follow the InAs-related PL intensities to sufficiently high temperatures (low values of $1000/T$) as a result of the inferior confinement afforded by the air/oxide/InAs/InP structures when compared with InAs quantum wells with InP barriers. Also, the lack of sharp "knees" in the plots of $\log(I_{PL})$ against inverse temperature, suggests a degree of temperature-dependent non-radiative recombination within the wells themselves.

The dependance of PL peak position on well width rules out the possibility that the surface layer is in fact an $\text{InAs}_x\text{P}_{1-x}$ alloy. Comparing the PL spectra of Figure 5 measured within one week of the samples being grown with those of Figure 6, which were measured twelve months after growth; we find a decrease in intensity combined with a shift of the peak positions to higher energies. This is precisely the kind of behaviour one would expect as oxidation decreases the effective width of the InAs quantum wells and so increases the confined state transition energies.

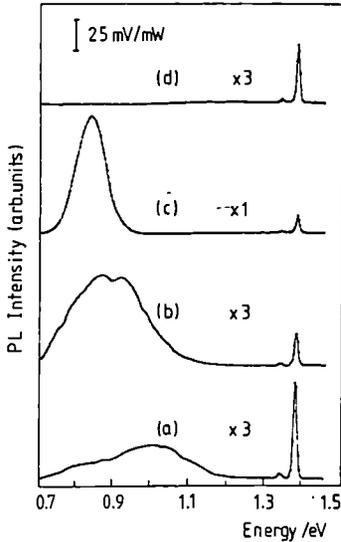


Figure 5: 10K PL spectra, acquired within a week of sample growth, for (a) 2.5 Å InAs/InP, (b) 5 Å InAs/InP, (c) 10 Å InAs/InP and (d) 20 Å InAs/InP.

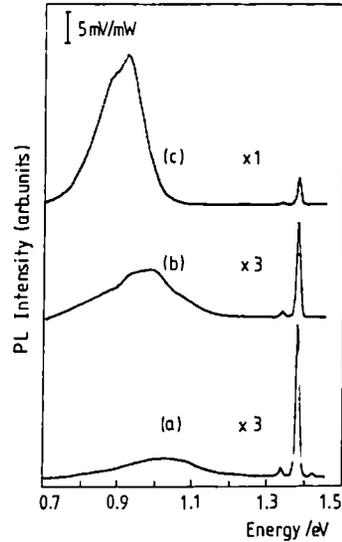


Figure 6: 10K PL spectra, taken 12 months after samples were grown, for (a) 2.5 Å InAs/InP, (b) 5 Å InAs/InP and (c) 10 Å InAs/InP.

CONCLUSIONS

We have reported the first observation of PL from InAs surface quantum wells grown on InP by MBE. Since radiative recombination within the InAs layer can be distinguished from PL arising from both bulk and surface defects, this system allows us to monitor the stability of both the InAs/InP and InAs/air interfaces via their influence on the InAs quantum well luminescence. These InAs structures still exhibit efficient carrier capture after 10 months exposure to air, although decreases in the PL emission intensities, coupled with the emergence of higher energy PL components, reflect the continued oxidation of the InAs surface with time. Those observations suggest the promise of using surface quantum wells as sensors as well as emitters [1]. This work was funded under the ESPRIT program of the European Economic Community, Basic Research Action grant no. 3086. Z.S. would also like to acknowledge the financial assistance of the U.K. Science and Engineering Research Council.

References

1. R.M.Cohen, M.Kitamura and Z.M.Fang, *Appl. Phys. Lett.* 50, 1675 (1987).
2. J.M.Moison, K.Elcess, F.Houzay, J.Y.Marzin, J.M.Gerard, F.Barthe and M.Bensoussan, *Phys. Rev. B* 41, 12945 (1990).
3. T.C.Hsieh, T.Miller and T.-C.Chiang, *Phys. Rev. Lett.* 55, 2483 (1985).
4. G.J.Davies, R.Heckingbottom, H.Ohno, C.E.C.Wood and A.R.Calawa, *Appl. Phys. Lett.* 37, 290 (1980).
5. D.I.Westwood, D.A.Woolf and R.H.Williams, *J. Cryst. Growth* 98, 782 (1989).
6. O.J.Glembocki and H.Pillar, in *Handbook of Optical Constants of Solids* (ed. Edward Palik, Academic Press, 1985) pp.503-516.
7. W.Seifert, J.O.Fornell, L.A.Ledebo, M.E.Pistol and L.A.Samuels, *Appl. Phys. Lett.* 56, 1128 (1990).

8. T.Y.Wang and G.B.Stringfellow, J. Appl. Phys. 67,344 (1990).
9. R.P.Schneider and B.W.Wessels, Appl.Phys.Lett.57, 1998 (1990).
10. H.Banville, E.Gil, A.M.Vasson, R.Cadoret, A.Tabata, T.Benyattou and G.Guillot, Proc. of SPIE Int. Conf. on Physical Concepts of Materials for Optoelectronic Device Application, Aachen, Germany (1990).
11. J.Singh and K.K.Bajaj, J. Appl. Phys. 57, 5433 (1985).
12. J.M.Moison, K.Elcess, F.Houzay, J.Y.Marzin, J.M.Gerard, F.Barthe and M.Bensoussan, Phys. Rev. B41, 12945 (1990).
13. J.D.Lambkin, D.J.Dunstan, K.P.Homewood, L.K.Howard and M.T.Emeny Appl. Phys. Lett. 57, 1986 (1990).

InAs/InP STRAINED QUANTUM WELLS GROWN BY HYDRIDE VAPOR PHASE EPITAXY AND STUDIED BY PHOTOLUMINESCENCE

J. Leymarie, M. Mihailovic, E. Gil, N. Piffault, A. Vasson,
A.M. Vasson, A. Tabata*, T. Benyattou*, G. Guillot* and
R. Cadoret.

Laboratoire de Physique des Milieux Condensés, Unité de Recherche Associée au
Centre National de la Recherche Scientifique n° 796, Université Blaise Pascal-
Clermont II, F-63177 Aubière Cedex, France.

(*) Laboratoire de Physique de la Matière, Unité de Recherche Associée au Centre
National de la Recherche Scientifique n° 358, Institut National des Sciences Appli-
quées de Lyon, 20 avenue Albert Einstein, F-69621 Villeurbanne Cedex, France.

SUMMARY

The excellent electronic and optical properties of indium-based compound semiconductors enhanced by strain have made them increasingly important to the realisation of long-wavelength optical and high speed electronic devices. In particular the InAs/InP system looks promising for the achievement of strained-layer quantum well lasers. Single and multiple strained InAs/InP quantum wells are grown by hydride vapor phase epitaxy which presents advantages as the control of the surface stabilization or the independence of gaseous species. The ultra-thin layers present a regularity of interfaces and a high degree of carriers confinement revealed by photoluminescence investigations.

1. INTRODUCTION

III-V strained and unstrained superlattices attract significant attention due to their device potential. They open new possibilities for band structure engineering and applications in the field of microelectronic and optoelectronic. One of the most interesting applications is found in strained-layer lasers because these structures can present a low in-plane effective mass at the valence band maximum thus providing a reduced carrier density for population inversion and a diminution of loss mechanisms such as Auger recombination and intervalence band absorption (1). A drastic reduction in the threshold current density is then observed and high quantum efficiency, high power output lasers emitting at pre-determined wavelengths have now been realized (2-4). Furthermore, the enhanced non-linear properties of strained-layer structures are of prime interest for the realisation of optical modulators and optical switching devices (5-7).

The $\text{In}_x\text{Ga}_{1-x}\text{As}/\text{InP}$ system is of great interest because strain can be investigated on a large scale (from -3.8 % for $x=0$ to 3.2 % for $x=1$). The effective electron mass decreases together with an increase of the peak velocity when the In fraction is increased from the lattice matched composition ($x=0.53$). (Ga,In)As materials with a rich In fraction look promising for the fabrication of high electron mobility transistors for high frequency applications.

Consequently, the achievement of such heterostructures places even higher demands on crystal growth techniques. Among the various techniques, the hydride vapour phase epitaxy (H.V.P.E.) in use in our laboratory offers some advantages: it allows the complete independence of the gaseous species carrying the III and V

elements to the substrate. Thus we have been able to study the role of each partial pressure and we have elucidated the decomposition reactions occurring in the vapour phase (8). For example, in a hot wall reactor such as a H.V.P.E. one, the high rate of hydride decomposition reduces the waste of reactants and the amounts of poison gas. In addition, most reactions being at equilibrium, the determination of the limiting surface reactions controlling the growth rate is easier than in cold wall reactors. The achievement of InAs/InP quantum wells in our laboratory and of quantum wires by Cox et al. (9) or studies of regrowth on InP etched mesas (10) prove the interest of H.V.P.E. for low dimensional structures growth.

Our purpose is to investigate the optical properties of InAs/InP quantum wells (QW's) and multi-quantum wells (MQW's). Although this system has been forecast to be the optimal valence-band structure for long-wavelength strained-layer quantum well lasers (11), little work has been published about the growth and characterization of InAs/InP low dimensional structures (12-14).

In this paper, we present a study of ultra-thin InAs/InP strained quantum wells. The optimum growth conditions for the fabrication of thin layers with smooth interface morphology has been determined with the help of photoluminescence investigations.

2. EPITAXIAL GROWTH

The HVPE method supposes that the vapor phase is composed of arsine AsH_3 and phosphine PH_3 , as well as the chloride species InCl and GaCl . These last species result from reactions of HCl flow above indium and gallium metallic sources at high temperature, so that HVPE is a hot wall method. Experiments are performed in an open flow reactor under atmospheric pressure.

From bulk material to heterostructures

The achievements of heterostructures requires three main steps: the knowledge of growth mechanisms and rates for bulk materials, the study of transient stages in order to have good interfaces, the accurate control of growth and transient times sequences. First, we briefly describe these steps.

- Throughout previous theoretical studies and numerous experiments on epitaxied InP and (In,Ga)As bulk materials, the parameters such as temperature and partial pressure of the various gas in the reactor that govern the composition and the growth rate have been investigated. Published work about mass spectroscopy experiments and computation of flow patterns (15) are used to ascertain our knowledge of the gaseous phase composition above the substrate. From these results, we postulate several growth mechanisms to which we apply Eyring's kinetic theory, involving an activated complex (16). In order to choose between these mechanisms we calculate the growth rate using some well-known thermodynamic data and compare the direction of variation of theoretical graphs to experimental ones. It leads to results relevant to various types of reactor for hydride method epitaxy. In addition, the accurate fitting of the curves gives the values of thermodynamic parameters which were not already known. Forcing them to remain in a plausible range provides a further test for the theory. These accurate values are necessary to determine the partial pressures and the temperatures for reproducible experiments in our reactor.
- The growth of low dimensional structures requires short growth time sequences of a few seconds. Interface quality depends on a sharp determination of growth periods

by controlled transient phases. Transient periods leaving the InP substrate under hydrogen or hydride flows only have already been used (17). Moreover, previous studies of the limitation of growth rate by chloride atoms desorption (16) suggested the use of this self limited mechanism to stabilize the surface of the sample under a chloride flow (18). For each type of structure to be grown, the choice between the three types of transient stages (hydrogen, hydride and chloride flow) results from a balance between a good stabilization and a simplification of the changes of flows above the substrate. For InP/InAs/InP growth sequences, we choose to keep the InCl flow constant all along the experiment since it is the common compound of InP and InAs growth.

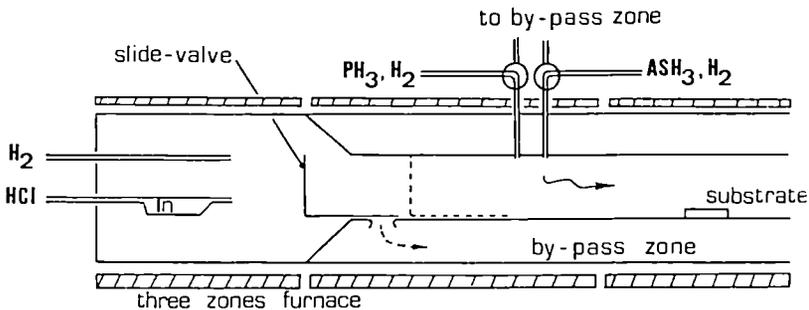


Fig. 1: Simplified sketch of the reactor. Hydride flows are pre-established with the help of three-ways valves. Source flows are exhausted to the by-pass zone while InCl flow is prepared (dashed position of the slide-valve). For InP and InAs growth, source flows are released to the deposit zone (solid position of the slide-valve).

- The complexity of tasks in heterostructure achievement has required an improvement of our experimental set up. The main idea for the design of the reactor has been the following: to provide its own tube to each gas and even to each function of the gas. Most of the tubes are equipped of three-ways valves in order to pre-establish the flows and derive the gas into the by-pass zone when it is not in use. Automation of the valves driving has been achieved for a better reproducibility of experiments.

Experimental process

All samples are Fe-doped or S-doped InP (100) substrates with a 3° misorientation towards $\langle 011 \rangle$. The surface is chemically cleaned before the introduction into the reactor under N₂ atmosphere. The InP is first heated up to the growth temperature (650°C) under phosphine atmosphere whose PH₃ partial pressure is of 2×10^2 Pa.

- **InP buffer growth:** It starts with the introduction of InCl flow. Partial pressures calculated at thermodynamic equilibrium are given in figure 2. The flow rate over the substrate is 20.5 cm s^{-1} for a total H₂ carrier gas flow of 3 l min^{-1} . The InP growth rate in the regime of homoepitaxy is of $9 \mu\text{m h}^{-1}$. A 8000 Å buffer is grown.
- **First transient:** The InP buffer growth is stopped by turning off the PH₃ supply valve and sweeping the PH₃ line with H₂. For 10 seconds, the InP buffer layer surface is kept in the chloride atmosphere (see partial pressures figure 2). The InP surface is stabilized under the InCl partial pressure and the residual HCl produced by the indium source whose efficiency is of 94%. Chloride stabilization blocks the P atoms

evaporation (18). The HCl initial flow on the In source is chosen taking into account the residual HCl flow, in order to settle the InP growth and also to stabilize the InP buffer surface. The HCl residual flow, depending on the HCl initial quantity and the flow rate above the In source, is calculated to prevent deposit from In_xCl_y species or etching of the InP surface. The time of 10 seconds is sufficient to totally evacuate the reaction chamber from PH_3 trace so that, an AsH_3/PH_3 recovering, leading to $\text{InAs}_y\text{P}_{1-y}$ (19) is avoided.

- **InAs growth:** It starts by turning on the AsH_3 valve. Depending on our experimental conditions (geometry, temperature, AsH_3 flow rate) AsH_3 is totally decomposed. Using a thermodynamic equilibrium approach, the partial pressures may be calculated (given in figure 2). InAs growth is interrupted when the AsH_3 valve is turned off. To control the geometry of the AsH_3 front, AsH_3 is carried with a total high H_2 flow that is kept constant during each step of the growth cycle, even if AsH_3 supply is disconnected. High flow rate in the inlet tube is adjusted to quickly sweep the AsH_3 gas and to control the spreading by diffusion of the initially abrupt pressure front. Each switching transient phase is calculated in order to minimize partial pressures as well as gas speed variations.
- **Second transient:** The InAs surface is kept under InCl (HCl) atmosphere (the same as for InP stabilization). During this step, a InAs surface is etched so that the InAs layer thickness is determined by a growth time t_g and by an etching time t_e . Previous work (20) where t_g was varying from 4s to 24s followed by etching times of 4s to 14s, allowed us to model the InAs thickness e_{InAs} (monolayers) as follow:

$$e_{\text{InAs}} = R_g(t_g + \tau_g) - R_e(t_e - \tau_e) \quad (1)$$

where R_g and R_e are respectively the growth rate and the etching rate in monolayer/second (ML s^{-1}). τ_g and τ_e represent time constants fitting diffusional spreading of abrupt fronts during convective transport in the reactor as well as stagnant valve volume purge times.

Thanks to high resolution transmission electron microscopy measurements performed on a 14 monolayers InAs layer (21) and to a preliminary photoluminescence study where thickness of the QW is deduced from the related emission energy, the time constants are determined. Observations of InAs thickness evolution performed with short growth times and short etching times lead us to distinguish the first monolayers growth from the established regim that follows. Because of the difference of adsorption energies between InP and InAs surfaces, the growth rate is increased for the first InAs monolayer. In the same way, the etching phase is studied with a run of samples where t_g is kept constant to 6 seconds and t_e is varying from 9 to 15 seconds. The effective etching phase seems not to start immediately, so that the growth rate and the etching rate are re-adjusted for the thinner layers presented here. The final empirical equation is then:

$$e_{\text{InAs}}(\text{ML}) = 1.0(t_g + 1.2) - 0.5(t_e - 1.0) \quad (t_g \text{ and } t_e \text{ in seconds})$$

- **Cap layer:** Finally, a 1000 Å InP cap layer is grown by turning on the PH_3 valve. The sample is then cooled down in the same PH_3 atmosphere as that for the heating up. As an introduction to the multilayers growth study, the $t_g = 6\text{s} / t_e = 9\text{s}$ sample has been annealed at growth temperature. The idea is to test the behavior of the first QW's during a multilayer growth. The first quantum well of a 10 or 20 periods superlattice is kept respectively 5 minutes and 10 minutes at growth temperature before cooling down.

A first part of the 6s / 9s QW has been submitted to a 5 minutes annealing (accounted from the growth temperature stabilization after the heating up) under PH_3 atmosphere (2×10^2 Pa). A 10 minutes annealing has been then carried out on the second part of the sample.

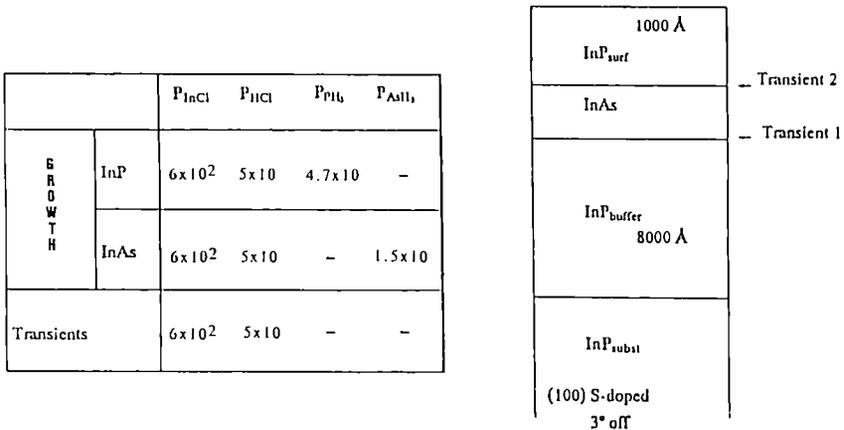


Fig. 2: In the table on the left, partial pressures are expressed in Pa. InP/InAs/InP heterostructure is schematically drawn on the right side.

For multilayer growth, the sequence previously described is repeated until the desired number of InAs well/InP barrier couples is reached. The choice of chloride transients is of particular interest for multilayers growth, because of a simplification of the changes of flows above the substrate. A severe control of the sharpness of interfaces is a guarantee for an accurate interpretation of photoluminescence spectra, especially when transient phases are numerous (up to 20 for a 10 InAs / InP QW's structure). As disturbances in the balance of flows only affect elements V flows (InCl is never stopped), variations of pressure and flow rate are controlled with the use of H_2 counterflows. A 10 InAs/InP periods structure has been processed where InAs layers were performed with the $t_g = 6\text{s} / t_e = 9\text{s}$ sequence, separated by 200 Å InP barriers.

3. PHOTOLUMINESCENCE STUDY

Photoluminescence (PL) experiments have been performed from liquid helium to room temperature. Samples are placed in a continuous flow cryostat. PL is excited by the 5145Å line of a CW argon-ion laser, and analysed by a 0.64 m focal length monochromator. The laser beam is chopped about 17 Hz and the induced PL signal is detected by a liquid N_2 cooled germanium detector and amplified using a conventional lock-in system.

The discussion will be limited to samples labelled 2 ($t_g = 6\text{s}$, $t_e = 11\text{s}$) and 3 ($t_g = 6\text{s}$, $t_e = 9\text{s}$) and the multiquantum well structure ($t_g = 6\text{s}$, $t_e = 9\text{s}$ for each InAs layer). These samples were made during a same run where the growth time was kept constant.

PL spectra of three samples are displayed in figure 3. The high energy part of the spectrum corresponding to sample 1 concerns the InP luminescence. The well-known transitions are the recombinations of free excitons or excitons bound to neutral donor

impurities labelled as band edge excitons (bee) and the neutral donor-acceptor pairs recombination (D^0A^0).

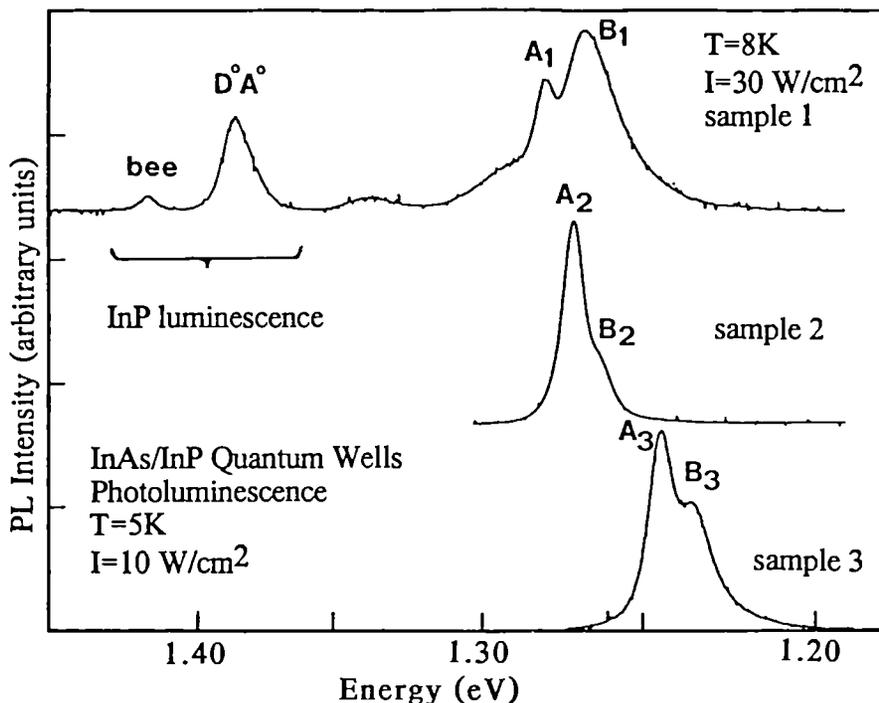


Fig. 3: 5K photoluminescence spectra of InAs/InP quantum wells. The InAs thickness is expected to be respectively 2 ± 1 ML and 3 ± 1 ML for 1.27-1.28 eV (samples 1 and 2) and 1.24 eV (sample 3) lines.

The common feature of the luminescence spectra of the InAs quantum wells is the existence of two lines (A and B) separated by about 10 meV. The A-line is relatively narrow (the full width at half maximum (FWHM) varies from 7 to 9 meV under the moderate injection conditions); it is attributed to the excitonic recombination of electrons and heavy holes confined in the strained InAs layer. The energies of the A_i lines ($i = 1, 2,$ and 3 for samples 1, 2 and 3) which are respectively 1.2787 eV, 1.2704 eV and 1.2440 eV indicate that the width of QW's is within a few monolayers as discussed below. It is worth noting that the FWHM is quite dependent upon the excitation intensity. When the excitation of sample 2 is increased by two orders of magnitude, the linewidth varies from 6.0 to 9.0 meV but the energy of the maximum does not move significantly towards higher energies. The same behavior is observed in other samples. If the linewidth can be analysed in terms of uniformity of the wells, it can be concluded that the narrow emissions reveal a good homogeneity of the interfaces (12). The broadening of the A line under an increasing illumination can be explained by heating of the free-excitons population.

Considering now the evolution of the line intensities as a function of the excitation power (I_{exc}), it is found that the intensity of the A_i lines varies approximately as $(I_{exc})^{\alpha_i}$ where α_i is respectively equal to 0.80, 1.3 and 0.95 for $i = 1, 2$ and 3 . In the same

conditions, the B_i lines intensity presents a sublinear behavior. Moreover, it appears that between the investigated samples the α_1 coefficient is decreasing when the ratio of B_i intensity to A_i intensity is increasing (see figure 3). This result is consistent with the identification of the B band as emissions involving impurity levels. The greater is the impurities concentration, the more important is the dissociation of free excitons rather than their direct recombination. The low energy difference between the A and B lines indicates that the involved impurities are shallow and then present a low ionization energy. Photoluminescence as a function of temperature confirms this assumption.

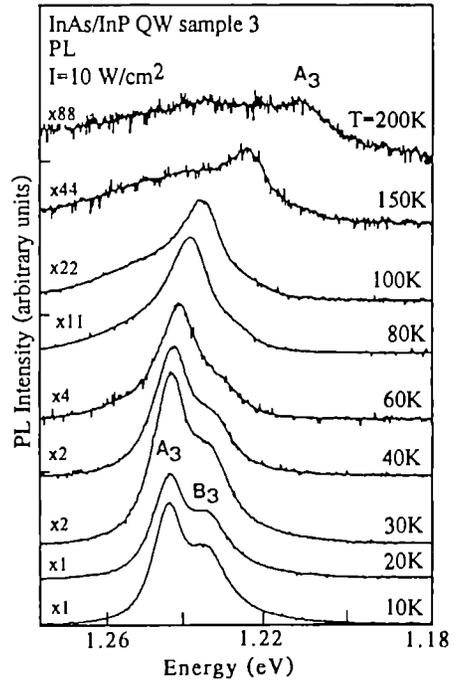


Fig. 4: Series of photoluminescence spectra for InAs/InP quantum well (sample 3) as a function of temperature from 10K to 200K. The quenching of B_3 line above 80K allows to attribute this line to impurities associated transitions. A_3 line is the excitonic recombination.

PL spectra of sample 3 are displayed in figure 4 where the temperature is varying from 10K to 80K. As a function of increasing temperature, the PL peaks continuously shift to lower energy due to the reduction of the band gap energy. No blue shift is observed as in the case for wide (In,Ga)As/InP quantum wells where this phenomenon is interpreted as the delocalisation of excitons bound to impurities, alloy fluctuations or well thickness variations (22, 23). The fact that above 80 K, B_3 (and also B_2) emission is not detectable is consistent with the assumption that this band involves impurity levels. In the same way, B_1 emission collapses and disappears over the same temperature range. The high energy shoulder observed above 100 K in sample 3 could be attributed to band to band transitions or emissions from excited subband levels. This feature is common to all the investigated samples and this shoulder is even detectable at low temperature under rather high excitation in sample 1 (figure 3).

The high energies of the InAs QW luminescence lines (1.270-1.279 eV) are very close to those recently obtained in InAs/InP QW's elaborated by organometallic vapor phase epitaxy (OMVPE) (12). In their work, the authors underline in particular that the energy and linewidth of the emissions are very sensitive to growth conditions and assign the

1.28 eV peak to the luminescence coming from one InAs monolayer. In our case, it is rather difficult to obtain such an ideal well because the misorientation of the (001) InP substrate (3° in the $\langle 011 \rangle$ direction) induces terraces of about 60 Å in length for monolayer steps (≈ 3.0 Å in height) which are much smaller than the excitonic radius (≈ 300 Å). Such a situation which could imply confinement in the layer plane suggests a discontinuous InAs monolayer. A calculation of the electron/heavy hole transition energy for an InAs monolayer sandwiched between two InP barriers gives a value of about 1.34 eV which is larger than that observed experimentally (24). It is also worth noting that the emission line associated to an InAs submonolayer (0.8 ML) in gallium arsenide is localized in energy just below the carbon related D^0A^0 emission band of GaAs (25). This result demonstrates the possibility to obtain very thin layers with a high energy of confinement. Consequently, we prefer to attribute respectively the 1.27-1.28 eV and the 1.24 eV peaks to 2 ± 1 and 3 ± 1 monolayers. The mean values of thickness are used to readjust the parameters of equation (1). Nevertheless, it is observed that the increase of the etching time up to 15s for a constant growth time (6s) does not induce a PL energy greater than 1.28 eV suggesting that even with long etching time, it seems impossible to completely etch the InAs layer under these experimental conditions. A kinetic approach is in progress to explain the fact that heteroepitaxied InAs on InP is stabilized when homoepitaxied InAs on InAs is etched. The difference of binding energies of InAs on InP and InAs on InAs probably induces this phenomenon.

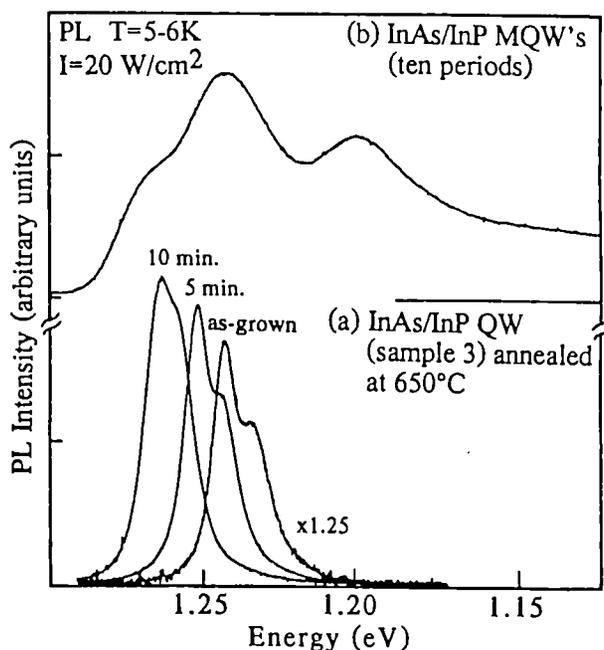


Fig. 5: (a) Luminescence of an InAs/InP quantum well (sample 3), first as-grown and then annealed (5 and 10 min. at 650C). The blue shift of spectra induced by annealing is attributed to the interdiffusion of V elements.

(b) Photoluminescence spectrum of ten InAs/InP quantum wells structure where each quantum well is grown in the same conditions as sample 3.
All spectra are recorded at 5-6K.

In order to analyse the optical properties of multi-quantum well structures, the annealing of a single quantum well is carried out at growth temperature for five and ten minutes. Low temperature ($T = 5\text{-}6\text{K}$) spectra of as-grown and annealed samples, given in figure 5a, show that the whole spectrum shifts to the high energy side with increasing annealing time. It is also found that the linewidth of the excitonic line increases from 9 to 14 meV. This result is attributed to the interdiffusion of P and As atoms at the barrier/well interfaces which changes the quantum well profile (26). Such a phenomenon has also been reported in annealed $(\text{In,Ga})\text{As}/\text{GaAs}$ strained quantum wells grown by low pressure OMVPE and In/Ga interdiffusion has been proposed to interpret the blue shift detected in PL measurements (27). In the particular case of ultra-thin quantum wells, the interdiffusion of isovalent species on a very short length (a few angstroms) is sufficient to appreciably modify the energy levels. The formation of a $\text{In}(\text{As,P})/\text{InP}$ QW by annealing can explain the luminescence energy shift and the broadening of the main line is attributed to the alloying effect on optical transitions.

The ten InAs/InP QW's structure presented here is grown under the same conditions as these of sample 3 and consequently it is not surprising to find that the maximum of emission corresponds to the luminescence maximum of the single QW (figure 5b). The broadening of the main line can be qualitatively explained by the effect of temperature on the first elaborated QW's at the beginning of the epitaxial growth (the growth time is about 5 minutes). The two lateral bands located at 1.27 eV and 1.20 eV are tentatively attributed to fluctuations of wells thickness. However, further work is required to elucidate this point.

4. CONCLUSION

This work evidences the feasibility of $\text{InP}/\text{InAs}/\text{InP}$ quantum wells and multi-quantum wells growth by the HVPE method. The heterostructures are studied by photoluminescence in order to compare them to similar OMVPE grown structures. Only surface InAs/InP quantum wells grown by MBE are reported to our knowledge (28).

The excitonic peak energy values obtained with HVPE grown and OMVPE grown quantum wells are approximately the same. Taking into account the approximations involved in the theoretical approach, we cautiously attribute the 1.28 eV peak energy to 2 ± 1 monolayers InAs thickness. The HVPE grown single quantum wells exhibit good quality interfaces as evidenced by 6 to 9 meV linewidths of excitonic emissions which are slightly narrower than the values reported in reference 12.

Concerning the multi-quantum wells grown by HVPE, the interdiffusion of V elements is probably responsible for the broadening of the photoluminescence spectra. Additional bands are present in multi-quantum wells grown by HVPE as well as in the single quantum well of about 4 monolayers grown by OMVPE. These bands are attributed to fluctuations of the InAs layer thickness.

In summary, the HVPE method appears to be comparable to OMVPE method as regards the quality of grown heterostructures. In addition, it is a cheaper and safer method, very promising for heterostructures achievement.

Acknowledgments: The authors would like to thank D. Castelluci for his technical contribution to this study. This work was supported by European Economic Community through ESPRIT Basic Research Action Contrat number 3086.

REFERENCES

- (1) For a review on valence band engineering in strained layer structures see: E.P. O'Reilly, *Semicond. Sci. Technol.* **4**, 121 (1989).
- (2) Ming C. Wu, N.A. Olsson, D. Sivco and A.Y. Cho, *Appl. Phys. Lett.* **56**, 221 (1990).
- (3) H. Temkin, T. Tanburn-Ek and R.A. Logan, *Appl. Phys. Lett.* **56**, 1210 (1990).
- (4) D.P. Bour, Ramon U. Martinelli, D.B. Gilbert, L. Elbaum and M.G. Harvey, *Appl. Phys. Lett.* **55**, 1501 (1989).
- (5) D.A.B. Miller, D.S. Chemla, T.C. Damen, A.C. Gossard, W. Wiegman, T.H. Wood and C.A. Burns, *Phys. Rev.* **B32**, 1043 (1985).
- (6) I. Bar-Joseph, G. Sucha, D.A. Miller, D.S.S. Chemla, B.I. Miller and V. Koren, *Appl. Phys. Lett.* **52**, 51 (1988).
- (7) A.J. Moseley, M.D. Scott, P.J. Williams, R.H. Wallis, J.I. Davies and J.R. Riffat, *Electronic Letters* **23**, 516 (1987).
- (8) M. Harrou, L. Chaput, A. Bendraoui, M. Cadoret, C. Pariset and R. Cadoret, *J. of Crystal Growth* **92**, 423 (1988).
- (9) M. Cox, D.E. Aspnes, S.J. Allen, P. Bastos, D.M. Hwang, S. Mahajan, M.A. Shadid and P.C. Morais, *Appl. Phys. Lett.* **57**, 611 (1990).
- (10) R.F. Karlicek, B.P. Segner Jr., J.D. Wynn, A.J. Becker, U.K. Chakrabarti and R.A. Logan, *J. of Electrochem. Soc.* **137**, 2639 (1990).
- (11) E. Yablanovitch and E.O. Kane, *IEEE J. Lightwave Technol.* **LT-6**, 1292 (1988).
- (12) R.P. Schneider, Jr. and B.W. Wessels, *Appl. Phys. Lett.* **57**, 1998 (1990).
- (13) M. Mihailovic, M. Cadoret, H. Banvillet, E. Gil and R. Cadoret, 5th Int. Conf. on the Physics of Electro-Optic Microstructures and Microdevices, Heraklion, August 1990. *Superlattices and Microstructures* **8**, 175 (1990).
- (14) H. Banvillet, E. Gil, R. Cadoret, P. Disseix, K. Ferdjani, A. Vasson, A.M. Vasson, A. Tabata, T. Benyattou and G. Guillot, *J. of Appl. Phys.* (1991) (to be published).
- (15) G.B. Stringfellow, *J. of Crystal Growth* **68**, 11 (1989).
- (16) R. Cadoret, *Currents Topics in Materials Science*, Ed. E. Kaldis, vol. 5, 220 (1980).
- (17) D. Gautard, J.L. Laporte, M. Cadoret and C. Pariset, *J. of Crystal Growth* **71**, 125 (1985).
- (18) M. Mihailovic, H. Banvillet, B. Gruzza, E. Gil and M. Cadoret, *Proc. of 3rd European Conference on Crystal Growth*, Budapest, May 1991.
- (19) R.P. Schneider, Jr. and B.W. Wessels, *Appl. Phys. Lett.* **54**, 1142 (1989).
- (20) H. Banvillet, E. Gil, A.M. Vasson, R. Cadoret, A. Tabata, T. Benyattou and G. Guillot, *Proc. of Int. Conf. on Physical Concepts of Materials for Novel Optoelectronic Device Applications*, Aachen, november 1990, vol. 1361, p972.
- (21) M. Pitaval, private communication.
- (22) D. Moroni, J.P. Andre, E.P. Menu, Ph. Gentric and J.N. Patillon *J. Appl. Phys.* **62**(5), 2003 (1987).
- (23) M.S. Skolnick, P.R. Tapser, S.J. Bass, A.D. Pitt, N. Apsley and S.P. Aldred *Semicond. Sci. Technol.* **1**, 29 (1986).
- (24) J. Camassel, J.P. Laurenti, S. Juillaguet, F. Reinhardt, K. Wolter, H. Kurz and D. Grützmacher, *J. Crystal Growth* **107**, 543 (1991).
- (25) R. Cingolani, O. Brandt, L. Tapter, G. Scamarcio, G.C. La Rocca and K. Ploog, *Phys. Rev.* **B42**, 3209 (1990).
- (26) T. Fujii, M. Sugawara, S. Yamazaki and K. Nakajima, *J. Crystal Growth* **105**, 348 (1990).

- (27) F. Iikawa, P. Motisuke, J.A. Brum, M.A. Sacilotti, A.P. Roth and R.A. Masut, *J. Crystal Growth* **93**, 336 (1988).
- (28) Z. Sobiesierski, S.A. Clark and R.H. Williams, to be presented in this conference.

FAULT INJECTION FOR THE EXPERIMENTAL VALIDATION OF FAULT TOLERANCE*

Jean Arlat, Yves Crouzet and Jean-Claude Laprie

LAAS-CNRS, 7, Avenue du Colonel Roche
31077 Toulouse Cedex - France

SUMMARY

This paper addresses the problem of the use of fault injection for testing algorithms and mechanisms implementing fault tolerance with respect to specific inputs they have been designed and implemented to deal with: the faults. First we present a global characterization of the fault injection attributes (the faults, the activation, the readouts and the measures) with respect to i) the various levels of abstraction of representation of the target system used in the development process (axiomatic, empirical and physical models) and ii) the validation objectives (fault removal and fault forecasting). Then we address more specifically the fault forecasting issues and proposes an evaluation method that implements the link between i) the analytical modeling approaches (e.g., Monte Carlo simulations, closed-form expressions, Markov chains) used for the representation of the fault occurrence process and ii) the experimental fault injection approaches (fault simulation and physical injection) characterizing the error processing and fault treatment provided by the fault tolerance algorithms and mechanisms. Finally, the last section depicts an approach for the characterization of the faults to be injected for uncovering design/implementation faults in the fault tolerance algorithms and mechanisms.

1. INTRODUCTION

The production of a dependable computing system heavily relies on various forms of hardware and/or software redundancies that are aimed at handling faults/errors, i.e. which embody the **fault tolerance** features of the system. A large number of both theoretical studies and experiments have shown that the adequacy and the efficiency — the **coverage** [Bouricius 69] — of the fault tolerance algorithms and mechanisms (FTAMs) have a paramount influence on the dependability and in particular on the measures (reliability, availability, etc.) usually considered for forecasting and estimating the level of dependability actually obtained.

As it applies to any validation activity, the use of formal verification methods and evaluation methods based on axiomatic models, enable a substantial improvement in the confidence level to be achieved. However, in spite of the continuous progress exhibited by these methods and due to i) the large number of uncertainties that still characterize the understanding of the behavior of computer systems in the presence of faults and ii) the difficulty in rating the parameters describing the fault tolerance actions, the complexity of such a task is generally too large to enable these methods to solve alone the validation problem. In that context, as is the case in any scientific processes, back-up from experimental methods is explicitly needed to get better insights in the underlying phenomena and hopefully try to address this challenge.

The realization of controlled experiments where the observation of the faulty behavior is explicitly induced by the introduction of faults into the system, i.e. the **fault injection** approach, has long since been recognized. Conceptually, fault injection can be seen as a means for verifying the FTAMs with respect to a particular class of inputs for the processing of which they have been specifically designed: the faults. Fault injection can be applied at various abstraction levels of representation of a target fault-tolerant system (FTS) [Arlat 90-a]: i) either on empirical models by means of fault simulation, (see e.g. [Segall 88, Czeck 90, Goswami 90]), ii) or on physical models (prototypes) by means of physical fault injection, (see e.g. [Gunnflo 89, Arlat 90-b]).

Although fault simulation and physical fault injection differ by i) the stage in the development process they can be applied, ii) the types of errors that can be induced by the techniques used for

* This work was performed within the framework of PDCS, ESPRIT Basic Research Action n° 3092, "Predictably Dependable Computing Systems".

injecting faults (logical vs physical) and iii) the controllability and observability they provide, they are both based on the design and realization of a **test sequence** made up of a series of controlled experiments; each **experiment** being characterized by the injection of a fault and the observation of the reactions of the target system to this stimulation. Moreover, increasing the confidence in the validation process usually requires a cooperation between these efforts; examples of cross-fertilization between fault simulation and physical fault injection are given by considering:

- the influence of fault simulation on physical fault injection: the simulation of internal faults may provide helpful data to calibrate the parameters used to characterize the faults injected in pin-level physical fault injection experiments [Choi 91],
- the influence of physical fault injection on fault simulation: the erroneous behaviors induced by physical fault injection experiments may provide useful data to elaborate fault/error models suitable for higher level fault simulation experiments [Segall 88].

The paper is made up of three sections that cover the fundamental aspects of the contribution of fault injection — encompassing fault simulation and physical fault injection methods — to the validation of the fault tolerance.

In order to provide a comprehensive and self-contained treatment of fault injection, section 2 is aimed at identifying the major fault injection attributes with respect to the complementary validation objectives: **fault removal** (how to reduce the presence of faults) and **fault forecasting** (how to estimate the present number, the future incidence and the consequence of faults) [Laprie 85].

Section 3 addresses more specifically the fault forecasting issues; for sake of conciseness, it only summarizes the main features of an experimental evaluation method aimed at bridging the gap between i) the analytical modeling approaches used for the representation of the fault occurrence process and ii) the experimental measures obtained by fault injection approaches characterizing the error processing and fault treatment provided by the FTAMs.

Up to now — except for the work related to the validation of fault tolerant protocols recently reported in [Echtle 91], most fault injection studies have focused on the fault forecasting objective. Although the data gathered during fault forecasting-aimed experiments can be used in practice in a feedback loop, to impact on the design/implementation of the FTAMs, we advocate that a more efficient and direct approach to this problem need to be investigated in using results already obtained in the test domain. Accordingly, section 4 deals with the fault removal issues by proposing and illustrating an approach for specifically determining the faults to be injected in order to uncover potential design/implementation deficiencies in the FTAMs that may impair their expected behavior when faced to faults they are intended to handle.

We are conscious that the material presented in the paper is of non-homogeneous depth. Indeed, the third section synthesizes a substantial amount of work carried out with respect to the fault forecasting objective while the fourth section exhibits mainly our preliminary critical views on the advantages and limitations in addressing more specifically the fault removal objective. In spite of this current unbalanced treatment, we strongly believe that a major reward can be obtained from a parallel investigation of these two objectives as is evidenced in section 2.

2. CHARACTERIZATION OF THE FAULT INJECTION ATTRIBUTES

2.1. Fault Injection and Validation Objectives

As already mentioned in the introduction, the fault injection-based validation of the FTAMs, addresses the two dimensions of validation: fault removal and fault forecasting.

With respect to the **fault removal** objective, fault injection is explicitly aimed at reducing the presence of faults in the design/implementation of the FTAMs whose consequences would be *deficiencies* in their expected behavior when faced to faults they are explicitly intended to handle: faults are injected to uncover such potential **fault tolerance deficiencies** faults (in short, **ftd-faults**) and accordingly, to determine the most appropriate actions to improve the FTAMs.

In the case of **fault forecasting**, the main issue is to rate the *efficiency* of the operational behavior of the FTAMs: this type of test can thus be seen primarily as a test of conformance of the FTAMs with respect to their overall behavioral specification in presence of faults. In practice, this corresponds to provide estimates for the parameters that usually characterize the operational behavior of the error processing and fault treatment coverage factors, dormancy, latency, etc.

Of course, these two dimensions actually cooperate together rather than compete in the process of validation of the FTAMs which is in fact a recursive validation process. The cooperation between these two dimensions actually enlightens the general concept of **coverage** [Laprie 85] that refers to a

measure of the representativeness of the situations to which a system is submitted during its validation compared to the actual situations it will be confronted during its operational life. In the context of a fault injection test sequence, this concept may apply to:

- the coverage of the test with respect to the activity of the FTAMs provided by the fault/errors injected,
- the coverage of the FTAMs with respect to faults they are specified to handle.

The complementary role of these dimensions can be further exemplified by considering the fact that when a deviation from the expected behavior is observed in the fault forecasting experiments, it is then usual to call for dedicated testing and diagnosis techniques in order to facilitate the search and elimination of the *ftd*-faults; however, this extrinsic fault removal capability corresponds only to a "by-product" and is in no way a systematic approach.

2.2. The Fault Injection Attributes

Figure 1 depicts the global framework that characterizes the application of fault injection for testing the FTAMs of a target system. The abstract description level enables the FARM (faults, activation, readouts and measures) attributes identified in [Arlat 90-a] to be further refined. For sake of clarity, it has been assumed that the FTAMs to be tested are imbedded in the target system but can be fully identified as a sub-system of the target system.

The figure shows only the information relevant to the testing of the FTAMs, i.e.: i) the error set *E* that actually controls their activity, ii) the error syndrome set *S* that characterizes the detection signals, alarms provided and reports on the fault tolerance actions taken.

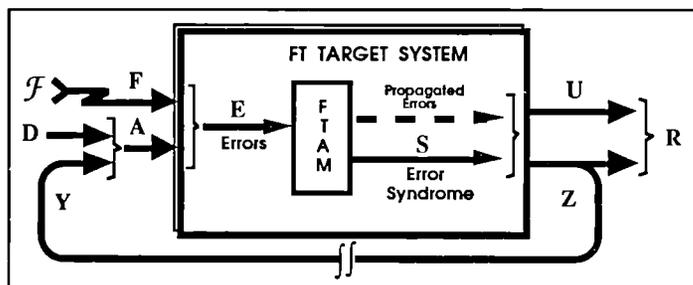


Figure 1 - The fault injection attributes

The *E* and *S* sets define the input and output domain for the verification/test of the FTAMs aimed at uncovering the potential (unknown) deficiencies in the FTAMs (the *ftd*-faults). An example for the *E* set is single error for a SED code; the associated typical example for the *S* set is parity error. These sets provide only minimal information that is in practice not sufficient to conduct a fault injection test sequence. Two main reasons require that they be refined or extended.

First, the fact that the direct injection of the elements of the *E* set might not be always feasible in practice or the consideration of specific FTAMs such as self-test programs (that are aimed at both activating faults and processing errors) for example, require to take into account that the *E* set results from the combination of two sets:

- the set *F* of the injected faults; *F* is of course a subset of \mathcal{F} , the set of all potential faults of the target FTS excluding however the *ftd*-faults,
- the set *A* that specifies the activation domain of the target system (and thus of the injected faults as errors).

Second, it is worth noting that the set *S* may include elements (e.g., exception raised, call for maintenance) that span respectively over two wider sets:

- the set *Z* that defines the internal state of the target FTS,
- the set *U* that characterizes the service provided by the FTS to its users.

However, due to error propagation and latency problems, the simple observation of the *S* set explicitly delivered by the FTAMs is not always sufficient to verify their proper operation. Further evidence has to be gained by checking for the integrity of some specific state variables or even more globally by actually observing the service delivered by the target system to its users. This leads to extend the set *R* of the readouts collected for each fault injection experiment to characterize the behavior in presence of faults to account for larger subsets of both *Z* and *U* sets.

The introduction of the *Z* set further enables the *A* set to be decomposed into two sets:

- the *D* set that designates the external input data,
- the *Y* set that defines the set of (current) internal states.

Classically, the next state $z(t) \in \mathbf{Z}$ becomes the current state $y(t) \in \mathbf{Y}$, after an interval ∂t corresponding to the stabilization of the internal state or the period of the clock in a synchronous system, i.e., $y(t+\partial t) = z(t)$.

Finally, the measure set \mathbf{M} defines the experimental **measures** (e.g., latency estimate, fault dictionary entries, etc.) obtained by combining and processing the elements of the FAR sets.

The model of figure 1 embodies explicitly the notion of fault in the input domain $\mathbf{I} = \{ \mathbf{D} \times \mathbf{Y} \times \mathbf{F} \}$. It enables the target fault tolerant system to be described by means of a function f that relates the input domain \mathbf{I} to the output domain $\mathbf{O} = \{ \mathbf{Z} \times \mathbf{U} \}$. The behavior of the target system is thus described by a sequence of states. The impact of a fault vector $f(t)$ can be perceived when the fault is activated:

$$\forall t, \exists d(t) \text{ and/or } y(t) \text{ such that } f(d, y, f; t) \neq f(d, y, f_0; t).$$

where $f_0(t)$ is the vector "absence of fault". In order to simplify the expressions, we use the notation: $f[\alpha(t), \beta(t), \gamma(t)] \equiv f(\alpha, \beta, \gamma; t)$.

This activation corresponds to the deviation from the expected trajectory:

- either as an **(internal) error** when only the state vector \mathbf{Z} is altered:

$$f(d, y, f; t) = (z', u; t) \neq (z, u; t)$$

where $z'(t)$ denotes an internal state distinct from the nominal one $z(t)$,

- or as an **error affecting the service** when, as a result of a failure, the vector from \mathbf{U} also deviates from the specified service:

$$f(d, y, f; t) = \begin{cases} (z', u'; t) \neq (z, u; t) & \text{(a)} \\ (z, u'; t) \neq (z, u; t) & \text{(b)} \end{cases}$$

where $u'(t)$ denotes an output distinct from the nominal one $u(t)$.

The use of the variable t enables to account for a possible deviation in value and/or in time.

The role of the FTAMs is to prevent such a fatal deviation by providing controlled alternate paths that may be taken when an error is observed, i.e., when the internal state has been modified.

An important remark should be made here to further precise the state set \mathbf{Z} (resp. \mathbf{Y}) and thus the notion of error. Indeed, the evolution of a system does not depend at any time on all its internal states. This leads to make a partition of the state vector $z(t)$ that distinguishes the state vector $z_s(t)$ — that characterizes the state variables that are sensitized at time t , i.e. the internal variables that actually impact on the evolution of the system at time t — from the state vector $z_d(t)$, that characterizes the variables that are not sensitized at time t . Such a distinction is useful in practice to account for latent errors (e.g., bit flips in a memory device induced by a transient fault):

$$f(d, y_d, y_s, f; t) = (z'_d, z_s, u; t) \rightarrow f(d, y'_d, y_s, f_0; t) = (z'_d, z_s, u; t)$$

Furthermore, such a distinction is essential:

- to characterize the fault-error-failure pathology: as dormant faults may not create erroneous behaviors, all erroneous states do not necessarily cause a failure,
- to design and implement the FTAMs: it is not necessary to try to observe nor to recover all system's states.

In the fault injection framework, it is worth noting that:

- the first point has a direct impact on the controllability for the definition of the fault/error injection method to cover the \mathbf{E} set and on the observability, in particular with respect to the control of the activation of the injected faults as errors and of the mutations induced by the propagation of these errors,
- the second point is especially important with respect to the observability of the response of the FTAMs in presence of faults.

Finally, the partition of the state set \mathbf{Z} allows to separate the notion of erroneous behavior in the target FTS (that may lead to the occurrence of a failure) characterized by $z'_s(t) \neq z_s(t)$, from the input domain of the FTAMs (the \mathbf{E} set). In particular, although only the $z'_s(t)$ states can be detected and processed by the FTAMs, it has to be verified that \mathbf{E} covers properly the $z'_s(t)$ states. More generally, this raises the problem of the coverage of the sets describing the behavior of the target system by the FAR attributes; the main problems concern:

- the coverage of \mathbf{F} with respect to the potential fault set \mathcal{F} ,
- the coverage of $\{ \mathbf{Z} \times \mathbf{U} \}$ by \mathbf{R} ,
- the activation coverage of the FTAMs provided by \mathbf{E} ,
- the mapping of $f(\mathbf{F}, \mathbf{A})$ into \mathbf{E} .

The first problem is very general and is inherent to any simulation approach; it is really dependent on the fault/error injection technique used and its global solution would require the development of a hierarchy of coherent fault/error models as proposed e.g. in [Joseph 88]. Although the treatment of this problem is beyond the scope of this paper, it will be further discussed — albeit partially — in section 3 along with the second coverage problem. The third and fourth problems will be specifically addressed in section 4.

2.3. Validation Objectives and Fault Injection Attributes

The table of figure 2 summarizes the main influences of the validation objectives on the fault injection attributes.

	(FTD)-FAULT REMOVAL	FAULT FORECASTING	
E	Error patterns activating the FTAMs: <ul style="list-style-type: none"> - specific faults deduced from fault assumptions made at design phase - test program to favor the activation of the injected faults and the propagation of the errors to the FTAMs (set Y) 	Large number of faults statistically distributed among the possible faults (set \mathcal{F})	F
		Input patterns simulating the operational utilization profile (set D)	A
R	Event occurrences and/or timing measurements (set S) (+ diagnosis data)	Event occurrences and associated timing measurements (sets U and Z)	R
M	Binary state characterized by the verification (or not) of the conjunction of a set of predicates (event occurrences or timing measurements)	Statistics on state occurrences, characterized by predicate combinations and time into and between states	M

Figure 2 - Impact of the validation objectives on the FARM attributes

2.3.1. The E, F and A sets

For fault removal, the maximum efficacy of the test is obtained when each experiment corresponds to the application of an error pattern in the E set, i.e. each experiment actually activates the FTAMs.¹ Two main approaches can be applied towards this goal:

- accelerate the activation of the faults in the F set and their propagation to the FTAMs interface by means of an A set elaborated by using the results of the structural testing theory as was applied in the experiments reported in [Crouzet 82],
- elaborate *a priori* the elements of the E set from:
 - either, the combined analysis of predetermined F and A sets in order to identify the minimal set of error classes that they generate (see e.g. [Raytheon 78]),
 - or, the analysis of the functional specifications of the FTAMs and/or of their structural description for the identification of the potential faults (fid-faults) that may affect their design/implementation.

The first approach is mainly empirical and the resulting coverage of the E set is mostly statistical. This may require a large number of experiments to obtain a significant test. Furthermore, as the experiments must be reproducible to enable the control of the design fixes made after the identification of a fid-fault, pseudo-random test sequences must be constructed.

In the second approach, the preliminary analysis usually enables the number of experiments to be reduced and the test sequence is deterministic². The first analysis method, is in general very complex and, in practice, it might be restricted to simple subsets of the \mathcal{F} set (e.g., permanent stuck-at-fault model on a limited size system). The second analysis is based on the functional analysis of the FTAMs and may benefit from some structural information on their architecture. However, although it might provide useful insight in this analysis, we feel that — due to their intrinsic limitations and to current

¹ However, in the case where the FTAM considered is a test program, the non activation of an injected fault actually reveals a fid-fault — a commandability deficiency — in the test program (e.g., see [Arlat 89]).

² The ability to directly apply the error patterns of E depends on the compatibility of the abstraction levels considered respectively for the description of the FTAMs and for the fault injection.

lack of knowledge concerning the ftd-faults — the application of (ftd-)fault-based test methods is still far from being practical. In section 4 we further investigate the main features of the method we propose for the elaboration of the E set.

For fault forecasting, the two major constraints are that the F set:

- corresponds to a statistical distribution among the possible fault set \mathcal{F} ,
- includes a **large number** of elements to achieve a significant confidence level.

As the goal of fault forecasting is to estimate the operational behavior of the FTAMs, the injected faults spread outside the strict hypothesis used for their design. The realization of a large set of experiments is facilitated by the fact that the volume of readouts to collect is significantly lower than in the case of fault removal (only statistical measures have to be elaborated, no diagnosis data has to be recorded).

The characterization of the A set is mainly based on the D set that should simulate the activity of the environment controlled/monitored by the system.

2.3.2. The R and M sets

For the R set, three major differences distinguish fault removal and fault forecasting:

- for fault removal, in order to carry out unit testing of the FTAMs, the R set might be restricted to the S set; however, for integration testing and for fault forecasting further information on the target system has to be taken into account (sets Z and U),
- although for fault removal, it is possible to devise separate tests with respect to distinct functional criteria:
 - either occurrence of specific events (e.g., fault activation, error detection, etc.),
 - or the duration of actions taken (e.g., error recovery, ...),
 for fault forecasting, it is generally required to account both for the occurrence of a significant event and for the instant of occurrence of this event,
- the amount of information to be collected in the case of fault removal is significantly larger due to the data recorded to help in the ftd-fault diagnosis and thus enabling a feedback to the design process.

In the case of fault removal, the M set is mainly characterized by a binary variable conditioned by the assertion or not of a conjunction of predicates issued from the specification of the FTAMs behavior. In the case of fault forecasting, the M set corresponds to statistical or probabilistic measures of:

- the existence of specific states corresponding to predicate combinations and of the distributions among these states,
- the sojourn time in the states and the time interval between the state transitions.

2.4. Characterization of a Fault Injection Experiment

A fault injection experiment is primarily controlled by the injection of a fault/error pattern in F/E, and by the observation of the resulting behavior of the target system.

Readouts collected in R during an experiment contribute to characterize the state of the target system by means of the assertion or not of a set P of predicates that are meant to abstract the specification of the behavior of the target system and thus of the FTAMs under test. Typical examples of predicates are: {fault_activated}, {fault_activated & error_signalled}, {error_signalled & proper service delivered}. Such predicates and/or their combinations define the set V of vertices of a graph that model the behavior of the target system (or of the FTAMs) in presence of faults. Such a graph can be either established *a priori* to describe anticipated behaviors or obtained *a posteriori* from the analysis of the R set, which is a form of model extraction from the experimental results (e.g. see [Choi 91]).

Figure 3 gives an example of such a graph. Transition 1 corresponds to the activation of a fault as an error and transition 2 identifies the cases when the injected faults are not activated, which may lead to non significant experiments. Transition 4 identifies the errors tolerated but not detected. Transitions 5 and 7 distinguish the cases of failure of the detection and tolerance mechanisms.

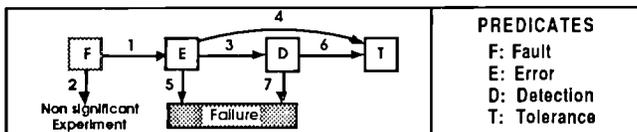


Figure 3 - Example of a predicate graph

Figure 4 further refines the characterization of predicates in \mathbf{P} and of states in \mathbf{V} obtained from the readouts in \mathbf{R} , in the case of a single binary predicate p ; three principal cases are accounted for depending on whether the predicate is expected or not:

- to maintain its value for all the interval $T = [t_F, t_M]$ that defines the observation domain for an experiment (figure 4-a),
- to change value once (figure 4-b) or more (figure 4-c) during the experiment.

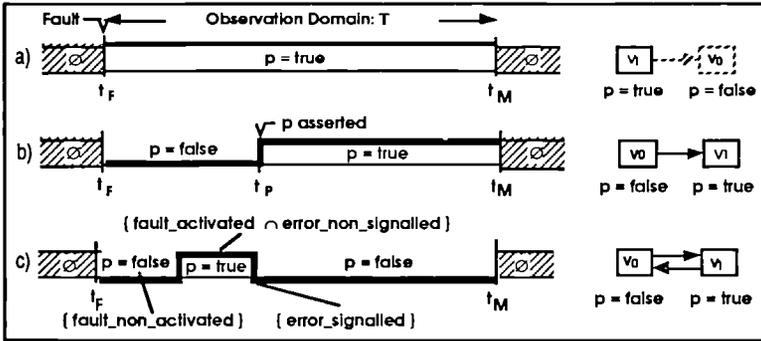


Figure 4 - Examples of tests of predicates

A typical example of the predicate corresponding to figure 4-a is the case of a predicate characterizing the **continuity of service delivery** in presence of fault (e.g., error masking) and accordingly rather characterizes reliability or availability features.

$p = \text{true} \Leftrightarrow \{ \text{acceptable_results_delivered} \}$ and $p = \text{false} \Leftrightarrow \{ \text{erroneous_result_delivered} \}$

A possible example for figure 4-b is constituted by the **testability** property for which an error must be signalled whenever a fault is present.

$p = \text{true} \Leftrightarrow \{ \text{error_signalled} \}$ and $p = \text{false} \Leftrightarrow \{ \text{error_non_signalled} \}$ (1)

Figure 4-c provides an example for the test of the **fail-safe** property defined as:

$p = \text{false} \Leftrightarrow \{ \text{fault_non_activated} \text{ or } \text{error_signalled} \}$ and

$p = \text{true} \Leftrightarrow \{ \text{fault_activated} \ \& \ \text{error_non_signalled} \}$

This corresponds to an alternating behavior between v_0 and v_1 that may be described by the decomposition of the predicate p into two elementary predicates of the types shown on figure 4-b:

$p = p_1 \ \& \ \bar{p}_2$ with $p_1 = \{ \text{fault_activated} \}$ and $p_2 = \{ \text{error_signalled} \}$

Moreover, the observation of the instant of assertion of a predicate is of paramount as it characterizes the temporal performance of the FTAMs tested; in particular for the predicate of figure 4-b, relation (1) can be modified to:

$p = \text{true} \Leftrightarrow \exists t_p \in T$

3. EXPERIMENTAL FAULT FORECASTING

3.1. Characterization of a Fault Injection Test Sequence

Figure 5 depicts the major interactions between the various levels of models used and a fault injection test sequence devoted to the fault forecasting objective.³

Based on the abstraction level considered for applying fault injection on the target system (and thus on the related fault/error models), the **injectable faults/errors** set can be deduced from the **possible faults/errors** set, i.e. the existing fault/error models. In particular, this specifies the level of application of fault injection (physical or simulation) and the capabilities of the fault injection device (permanent or transient faults, single or multiple faults, etc.).

³ This figure focuses on the **F** and **E** sets which constitute the fundamental attributes of fault injection. The impact of the **A** set is also important (in particular for characterizing **E**); it has not been mentioned here for better readability.

Further refinement for the derivation of the **considered faults/errors** for testing a particular target system, is usually obtained by considering *a priori* known data or field data specific to the application domain concerning for example the components, the architecture and possibly the environment of the target system (what multiplicity ?, are both transient and permanent faults to be considered ?).

As exhaustive testing among the considered faults/errors set is generally not feasible and in order to enable nevertheless a representative test, the **faults/errors** actually **injected** should be obtained either by determining equivalence fault/error classes or by statistical sampling in the considered faults set.

According to the output space of the **fault injection test sequence**, the collection of **significant readouts** in the R set depends of course on the abstraction level considered for the target system, but also on some *a priori* modeling of target system behavior in particular concerning the conditions for the observability of the tested FTAMs. These individual readouts may be used to react on the target system design.

More comprehensive experimental measures in the M set (coverage distribution, asymptotic coverage, mean coverage time, etc.) may also be elaborated from the statistical processing of the readouts assuming that the fault injection test sequence fits a suitable probabilistic and statistical model (e.g., see [Arlat 90-a] for a detailed description of these models and a definition of the main statistical measures). Depending on the **goal** of the fault injection test sequence, these models enable that, in relation with the number of faults/errors injected, **confidence levels** be associated to, either **confidence intervals** for the measures estimated or **hypothesis testing** concerning the value of an estimated parameter.

Finally, as the FTAMs coverage measures obtained are *conditional dependability measures* (i.e., conditioned to the occurrence of a fault), models incorporating the fault occurrence process are needed in order to enable the evaluation of **dependability measures** and the possible comparison with the assigned dependability objectives. Next paragraph, further investigates the interrelationship between the coverage experimental measures and the fault occurrence process.

3.2. Bridging the Gap between Analytical Modeling and Fault Injection

Figure 6 depicts the principal phases of analytical and experimental dependability evaluation that are respectively based on the construction and the processing of either axiomatic models (sequence 1-2-4-6), or empirical and physical models (sequence 1-3-5-7). It is worth noting that this simplified description is tailored to emphasize the most significant interactions (large arrows). Of course, both sequences may be used separately to impact the target system (e.g., parameter sensitivity analysis for early architectural design decision in the case of model-based evaluation or as a design aid for fault removal in the case of fault injection-based experimental test).

The transition from 2 to 5 depicts the necessary impact of modeling in defining the fault injection attributes (in particular with respect to the R set) and in conducting the test sequence. Transition 7-8 identifies two types of interactions:

- impact of models on experiments: the reference to the fault occurrence process, usually described in axiomatic models, is necessary to derive dependability measures,
- impact of experiments on models: i) characterization of the original models by calibrating their coverage parameters, ii) validation of the assumptions made in their elaboration and iii) refinement and possibly the partial extraction of the structure of models.

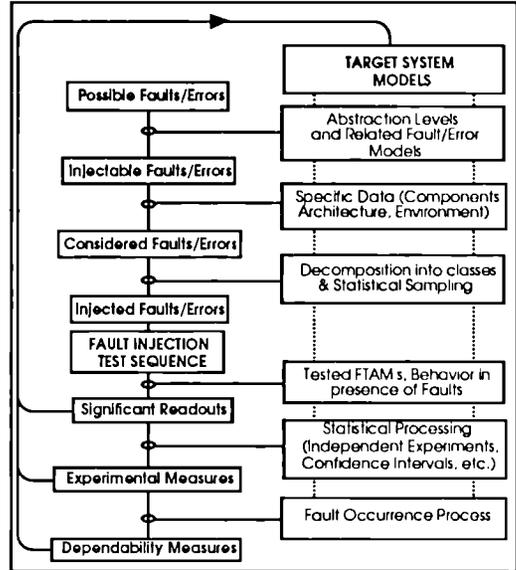


Figure 5 - Major interactions between fault injection and system modeling

Transition 2-5 depicts the necessary impact of modeling in defining the fault injection attributes (in particular with respect to the R set) and in conducting the test sequence. Transition 7-8 identifies two types of interactions:

- 1) models → experiments: the reference to the fault occurrence process, usually described in axiomatic models, is necessary to derive dependability measures,
- 2) experiments → models: i) characterization of the original models by calibrating their coverage parameters, ii) validation of the assumptions made in their elaboration and iii) refinement or partial extraction of the structure of models.

Processing of the models refined/extracted enables relevant dependability measures to be obtained. These provides a substantial basis for initiating feedbacks to improve the architecture of the target FTS. Next paragraph further analyzes the interactions induced by transition 7-8.

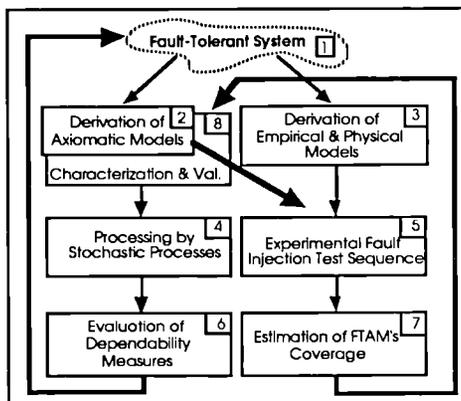


Figure 6 - Interactions between analytical and experimental dependability evaluation

3.3. Integration of Fault Process and Coverage Parameters

Three major classes of approaches can be considered to implement the integration between the fault occurrence process and the experimental coverage parameters: Monte Carlo simulation, closed-form expressions and Markov chains. Considering closed-form expressions, we have extended two types of models previously reported, although in different contexts; however, for easier tractability we have restricted our attention to asymptotic (constant) coverages.

The first model, adapted from [York 85], is based on a "canonical" decomposition of the reliability expression of a system; e.g., the reliability expression of a TMR system would write:

$$R_{TMR}(t) = R^3(t) + 3 \cdot C_1 \cdot R^2(t) \cdot \bar{R}(t) + 3 \cdot C_2 \cdot R(t) \cdot \bar{R}^2(t) + C_3 \cdot \bar{R}^3(t)$$

The usual ideal reliability expression is thus a particular case where the asymptotic coverage factors C_i associated to the error combination of order i are such that $C_1 = 1$ et $C_2 = C_3 = 0$.

The second model is based on the simulation work initially described in [Lyons 62]. It considers globally the accumulation of faults in the system to be a Poisson process and the failure process is considered to be monotonously related to the accumulation of faults. Basically:

$$\text{Prob.}\{ \text{FTS failure at } t \} = \sum_{k=1}^{\infty} \bar{C}_k \cdot \Phi_{Fk}(t) = \sum_{k=1}^{\infty} [\bar{C}_k \cdot \exp(-\lambda t) \cdot \sum_{i=k}^{\infty} \frac{(\lambda t)^i}{i!}]$$

where $\bar{C}_k = \text{Prob.}\{ \text{FTS failure} \mid \text{fault number} \geq k \}$ and $\Phi_{Fk}(t) = \text{Prob.}\{ \text{fault number} \geq k; t \}$.

These approaches have been investigated in detail in [Arlat 90-c]. Figure 7 summarizes their main merits and limitations.

	MONTE CARLO SIMULATION	CLOSED-FORM ANALYTICAL EXPRESSION		MARKOV CHAIN
		Canonical expression	Accumulation of faults	
Hypotheses	Decomposition into independent classes (modules)	Global Poisson process		Exponential distribution
Level of Integration	Potential <i>a priori</i> impact on the definition of the test sequence	<i>A posteriori</i> utilization only of the experimental measures to characterize the FTAMs parameters of the models		
Parameters considered	Asymptotic coverage essentially			+ Mean of the FTAMs coverage distribution
Experimental evaluation	Combines both fault occurrence and coverage processes	No specific feature	Customized test sequence	No specific feature

Figure 7 - Main characteristics of the integration approaches

Markov chains are especially attractive since they provide a tractable means to account for the main temporal characteristics of the coverage distribution as exemplified in the next subsection.

3.4. Calibration of the Coverage Parameters of a Markov Chain

Let us consider the model of figure 8-a that describes the behavior of a FTS. This model accounts for the coverage of the FTAMs with respect to the occurrence of a fault and the possible occurrence of a second (near-coincident [McGough 83]) fault while processing the first one.

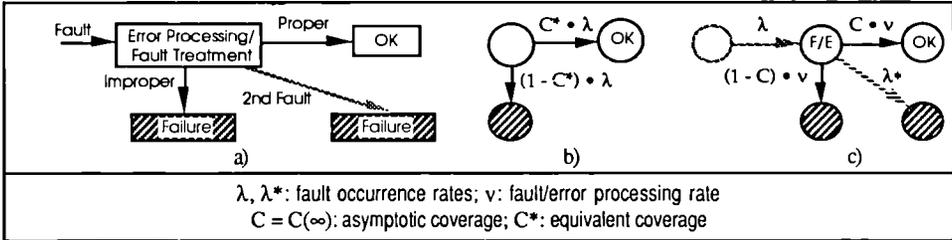


Figure 8 - Markov model of the coverage distribution

As shown in [Dugan 89] an equivalent Markov representation (figure 8-b) can be derived for such a behavior where the equivalent coverage C^* is defined as:

$$C^* \approx C (1 - \lambda^* \cdot E[T_d] + \frac{(\lambda^*)^2}{2!} \cdot E[T_d^2] - \dots) \tag{2}$$

where the constant parameter C can be identified as the asymptotic value of the coverage cumulative distribution distribution $C(t)^4$, λ^* is the rate of occurrence of a near coincident fault and the $E[T_d^i]$ designates the successive moments of the random variable defining the processing time of the FTAMs.

By limiting expression (2) to the first order and denoting $v = 1/E[T_d]$, the decision rate of the FTAMs, we obtain the model of figure 8-c. This model provides an essential "building block" to describe the coverage process, in particular for studying the impact of the temporal distribution.

From a practical point of view, the estimation of these parameters is penalized by the truncation of the observation domain T . Although, as shown in [Arlat 90-a], this leads to a conservative estimation of the asymptotic coverage, the estimation of the distribution of T_d is in practice more complex. More specifically, if $t_{pi} i = 1, \dots, N(T)$ denotes the realizations of the random variable T_p (time of assertion of the predicate p associated to the coverage of a specific FTAM in a test sequence) where $N(T)$ is the number of assertions in the observation domain $T = [0, T]$, then, $E[T_d] = E[T_p]/C$, and thus (2) can be simply expressed as:

$$C^* \approx C - \lambda^* \cdot E[T_p], \tag{3}$$

and as shown in [Arlat 90-c], $E[T_p]$ is (under)-estimated by the empirical mean of the conditional latency of the FTAMs (i.e. the empirical mean of the actually observed latencies):

$$\hat{E}[T_p] = \frac{1}{N(T)} \sum_{i=1}^{N(T)} t_{pi} \tag{4}$$

Figure 9 gives an example of a valued experimental graph (see figure 3). This graph was derived from the fault injection test sequence carried out in the framework of the validation of the fault-tolerant distributed architecture developed for the ESPRIT Precompetitive project Delta-4 [Powell 88].

The percentages indicate the values of asymptotic coverage for the predicates E (error), D (hardware detection) and T (tolerance by the communication protocol). The time measures indicate the means for fault dormancy and error

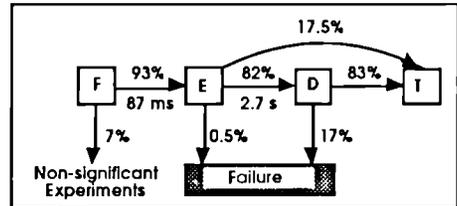


Figure 9 - Example of valued experimental graph

⁴ Other studies (e.g. [Geist 90]) rather refer to the probability density function of the coverage.

detection latency distributions; only asymptotic coverage was considered with respect to the T predicate. The detailed results of this test sequence are given in [Arlat 90-b].

4. FTD-FAULT REMOVAL

Unless the analysis of the readouts with respect to the latency lead to a decomposition of the experimental results into classes that can be related to the various detection mechanisms considered, in general, for the experimental evaluation of a FTS, it is not explicitly necessary to know the number and the type of the FTAMs.

On the contrary, for removal of fault tolerance deficiencies faults, the FTAMs to be verified need to be clearly defined and their interactions should be fully described.

Three major aspects have to be considered to conduct such a type of test (see figure 1):

- choice of the model(s) used to specify/describe the FTAMs to be verified,
- definition of the criteria with respect to which the test set E and S can be determined,
- implementation of the test sequence, i.e., generation of the E set from the combination of the F and A sets and of the R set from the U and Z sets.

4.1. Specification/Description Models

The level of representation of the target system is dependent on the types of ftd-faults that are to be revealed, i) faults in the definition of the FTAMs (e.g. in the case of complex algorithms) or ii) faults in their final implementation. Accordingly, ftd-fault removal can be useful at different phases of the production of a FTS and thus applied to models with different abstraction levels (axiomatic, empirical and physical models) [Arlat 90-a]. As usual, the earlier the potential deficiencies can be unveiled and removed, the better it is.

Irrespective of the abstraction level, models have to be established so that the attributes of the fault injection test sequence can be defined. Such a model specifying as precisely as possible the FTAMs to be verified is actually a prerequisite to perform any test sequence. Although we emphasize here the characterization of the E set, it is worth noting that these models should not only be used to specify the input patterns but also the associated outputs and thus define the observation domain.

Furthermore, the types of models to be used may vary significantly according to:

- the abstraction level of the target system (simulation model or implementation),
- the types of FTAMs considered (detection only or tolerance),
- the form of the specification available (formal or informal).

Potential candidate models thus include graphical or procedural expressions of the structure and/or behavior of the FTAMs. Typical examples are trees, Petri nets, flow charts, Pascal-like expressions of the specification (e.g., the Estelle and LOTOS languages [Diaz 89]), etc..

The differences in complexity of the FTAMs considered necessitate the use of different models: sophisticated fault tolerance procedures in a distributed system require the description of more complex error recovery mechanisms (timing, exclusion, concurrent evolution, synchronization, etc.) than those associated to simpler error detection mechanisms in a centralized system. Accordingly although a simple tree structure may suffice to describe the latter case, a Petri net description may be more suited to model the first one. Similarly, increased levels of complexity may also be found when addressing the verification of the integration of various FTAMs.

It is also worth noting that complementary models encompassing structural and functional descriptions are actually needed in practice. Indeed, a simple structural testing approach might not be sufficient since fault injection should not only be carried out to verify that the FTAMs properly process the errors in the E set but also that they are not activated in the absence of error (false alarms). As an example, in the case of a software algorithm, a structural testing approach is not always sufficient to reveal the absence of a specific path. It is then necessary to complement the testing by a functional testing approach.

4.2. Verification Criteria and Test Set Determination

When a formal specification of the FTAMs is available, it can be directly used to determine the test set given a specific test criterion. Depending on the form of the specification (procedural or graphical), typical test criteria may consist, for a program, of statement covering, branch covering, path covering or, for a Petri net, of related criteria defined on the state space described by the reachability graph, state covering, transition covering, sequence covering. These general criteria are thus independent of the FTAMs tested.

In practice, such a context enables all the methods and tools already existing in the domain of graph theory (e.g., see [Roy 70, Deo 74]) and structural testing (e.g., random testing [Thévenod-Fosse 91], symbolic execution [Camurati 88], fault-based algorithms [Kime 86], net analysis [Baumgarten 90] etc.) to be used to elaborate the test sequence.

The test sequence obtained at such a high level can be further used to perform a functional test for the subsequent simulations and implementations of the FTAMs.

It is worth noting that the selected criteria also define a precise basis to evaluate and compare the merits of the coverage provided by several test sets. In particular, this can be directly used for the determination of a test sequence with a specified coverage in the case of random testing.

4.3. Implementation of the Test Sequence

The main benefit that is anticipated from an early test of the FTAMs is to contribute to their validation in isolation which would enable their reusability in different systems. This would be especially true and useful for complex FTAMs (e.g., fault/error processing protocol for redundant software components in a distributed system). However, the test in isolation of some very simple detection mechanisms (e.g., a watchdog timer) might not be relevant, since its verification cannot be abstracted from the simulation of the F and A sets.

Furthermore, prototype-based fault injection will still be needed i) to verify the implementation including both hardware and software layers and ii) even in the case of simple FTAMs, as an integration test incorporating the interactions between various FTAMs. A major interaction to account for is the **dominance** effect, which corresponds to the case when the activation of a FTAM implies the activation of another one that preempts the consecutive error reporting/recovery actions.⁵

The application of the ftd-fault removal approach in the production process of a FTS system can thus be decomposed into two main steps (see figure 1):

- 1) identify the test patterns in the **E** set that properly activate the FTAMs; this step corresponds to a unit testing phase where the verification of the FTAMs is carried out independently of the FTS system in which they are imbedded,
- 2) verify the implementation of the FTAMs on a prototype; this step can be related to an integration testing phase.

It is assumed here that the first step is applied on a simulation model of the FTAMs to be verified, i.e. before a prototype is available; accordingly, the controllability (**E** set) and observability (**S** set) can thus be tailored to fit the description level of the model of the FTAMs. In the second step, reachability restrictions may significantly reduce the controllability and observability possibilities; it is thus necessary to determine the proper combination of the **F** and **A** sets suitable to obtain the **E** set characterized in the first step; this may be the case for example when, for practical reasons, faults can be injected only on the data bus of a prototype: it might be necessary to synchronize the injection with the activity on the bus to apply specific types of errors.

As in major realistic cases it is not possible to derive automatically (and reliably) an implementation from the verified simulation, the second step is aimed at removing the added implementation-specific ftd-faults. To illustrate the possible introduction of such a type of implementation ftd-fault let us consider a protocol aimed at processing faults/errors in replicated software components in a distributed system. Usually such a protocol is based on the assumption of atomic broadcast on the network enforced by underlying hardware self-checking components; however, for the analysis of the properties of the protocol at the simulation level this feature is assumed to hold and thus the validation only covers the logical principles — although non trivial — of the protocol. Implementation validation is thus mandatory to test and assess the actual coverage of the assumption (e.g., see [Arlat 90-b]).

This corresponds to the fact that some implicit or explicit assumption made in the definition of the algorithm simulated has been violated by the implementation. This second step thus corresponds to a test of the coverage of the fault/error models at the prototype level by the error models considered at the simulation level (i.e., the **E** set). This can be related to the issue of error confinement areas and layers in a FTS.

For both steps, the goal is to extract from the model (simulation or prototype), pertinent information to help construct the test sequence. In the next subsections, we illustrate the approach on two examples of FTS possessing several FTAMs either at the same layers or at different layers.

⁵ This can be related to the case of *invalidating faults* used in the theory of system diagnosis [Kime 86].

4.3.1. Single Layer FTAMs

Figure 10 gives an example featuring four error detection mechanisms: parity check, invalid op-code check, watchdog timer, memory protection violation check (invalid access).

Let us consider the case of the verification of the implementation of these mechanisms on a prototype when faults can only be injected on the data bus of the system and when the observability (S set) is restricted to the general error detection signal. The set E may be decomposed into several subsets, each corresponding to the errors activating one mechanism. The decomposition of the E set depicted on figure 10 shows that these subsets are not necessarily disjoint; e.g. an error on the data bus may either correspond to a parity error or create an invalid op-code. Accordingly, more than one mechanism may be activated at the same time with error processing times varying significantly (e.g. parity and watchdog). The problem then consists of determining the errors (test set) that sensitize only one of the mechanisms in order to be able to verify its proper operation. This may be achieved quite easily for the first three mechanisms:

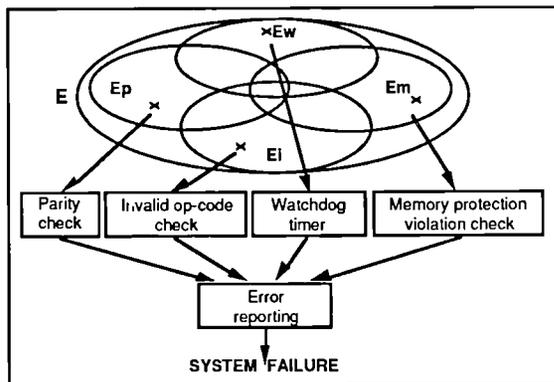


Figure 10 - Example of single layer FTAM

- parity check: inject errors on the parity bit ,
- invalid op-codes: force invalid op-codes with acceptable parity; in the absence of specific structural knowledge about the instruction decoder (functional test), it might be necessary to force all the invalid op-codes,
- watchdog timer: force a NOP (no-operation) code on the data bus with acceptable parity and measure the accuracy of the timer.

Such an isolation is however more difficult in the case of the mechanism detecting invalid access.

4.3.2. Multilayer FTAMs

Figure 11 describes the major layers of error detection identified in [Siewiorck 82].

LAYER	TYPICAL ERROR SOURCE	TYPICAL ERROR DETECTION TECHNIQUE
4-Application	Incorrect coding of algorithm	Reasonability checks
3-Operating System	Incorrect design	Consistency checks on data structures
2-Macrocode	Alpha particle flip bit	Memory protection violation
1-Microcode	Race condition	Error coding
0-Hardware	Environmentally produced transient	Replication

Figure 11 - Example of multilayer FTAMs

Let us assume that the patterns (E_i set) for testing the FTAMs of each layer i are identified. The test of the upper layers may possibly be carried out by simulation quite independently of the implementation. In order to test the implementation of the whole system and thus of each layer, for efficiency purpose it might be wanted to inject faults within one single layer, i.e. the lower layer.

However, for the testing of reasonability checks by injecting hardware faults be feasible it would be necessary that i) all the underlying layers be so inefficient that they let errors get through or that all the error detection mechanisms be deactivated, ii) the E_0 set be extended to activate all other upper layer's FTAMs ($E'_0 = E_{00} + E_{01} + E_{02} + E_{03} + E_{04}$, where E_{ij} designates the test patterns of layer i that activates the FTAMs of layer j and $E_{ij} = E_j$) and iii) that the errors applied (E_{0j}) cover all the classes in the E_j set previously determined. In practice this might not be always possible and thus in general it will be necessary to inject faults at different layers and for better efficiency at the layer corresponding to the E_j set associated to the FTAMs considered at layer j .

The identification of such layers has also an impact on the determination of the observability issues. This has mainly to deal with the problem of proper error confinement by the FTAMs. In practice, in the case of fault injection at layer i , it might be necessary to provide observation devices at layers $i+2$, $i+3$, etc. in order to complement the observation of the syndrome characterizing the detection mechanisms at layer $i+1$ and thus fully identify the erroneous behavior of the system. However, for fault removal, it might be sufficient to provide efficient observation devices at layer $i+1$ only, so as to check that errors are not propagated to the upper layers.

5. CONCLUSION

The paper has exposed a **unified** presentation of the main features of the application of fault injection in the validation process of fault tolerant computing systems accounting for both fault removal and fault forecasting objectives. The **refinement** of the fault injection attributes — the **FARM** sets — that has been introduced, provided a common framework that enabled to fully specify these objectives and to identify the specificities induced by each objective.

For **fault forecasting**, we have summarized the major points of an experimental dependability evaluation approach that actually implement the link between the statistical measures of the coverage of the fault tolerance obtained from a fault forecasting-aimed fault injection test sequence and the Markov models usually used to evaluate dependability measures. Our experience in the application of this method in the framework of the experimental validation of two real fault tolerant systems by means of pin-level fault injection with the tool MESSALINE [Arlat 90-a, Arlat 90-c], clearly demonstrated its usefulness with respect to the estimation of the coverage provided by the fault tolerance and also — to some extent — with respect to the removal of design/implementation deficiencies in the fault tolerance mechanisms.

Based on these achievements and on the limitations (late, indirect impact) of the feedback provided to the design process, we have presented here the main guide lines of an approach we are now proposing to complement the state of the art in the domain of fault injection-based experimental validation with respect to **fault removal**. The crux of this approach is to explicitly pose the problem of removal of deficiencies in the fault tolerance algorithms/mechanisms (FTAMs) in terms of a testing problem. The first step corresponds to the description of the FTAMs by means of a model suited to the complexity/type of the considered FTAMs. Then the inputs of the fault injection test sequence can be obtained from a set of criteria on the models by using the results of structural testing. Finally, the actual application of the test sequence significantly depends on the phase in the design process and thus on the abstraction level of the target system (simulation model or prototype).

ACKNOWLEDGEMENT

We would like to thank Alain Costes and David Powell for providing many helpful comments and suggestions on the work reported in this paper.

REFERENCES

- [Arlat 89] J. Arlat, Y. Crouzet and J.-C. Laprie, "Fault-Injection for Dependability Validation of Fault-Tolerant Computing Systems", *Proc. 19th IEEE Int. Symposium Fault-Tolerant Computing*, Chicago, IL, USA, June 1989, pp. 348-355.
- [Arlat 90-a] J. Arlat, M. Aguera, L. Amat, Y. Crouzet, J.-C. Fabre, J.-C. Laprie, E. Martins and D. Powell, "Fault Injection for Dependability Validation — A Methodology and Some Applications", *IEEE Transactions Software Engineering*, vol. 16, February 1990, pp. 166-182.
- [Arlat 90-b] J. Arlat, M. Aguera, Y. Crouzet, J.-C. Fabre, E. Martins and D. Powell, "Experimental Evaluation of the Fault Tolerance of an Atomic Multicast Protocol", *IEEE Transactions Reliability*, vol.39, n°4, October 1990, pp. 455-467.
- [Arlat 90-c] J. Arlat, "*Dependability Validation by means of Fault Injection — Method, Implementation, Application*", State Thesis Dissertation, National Polytechnic Institute, Toulouse, France, December 1990, also LAAS-RR n° 90-399. (In french).
- [Baumgarten 90] B. Baumgarten, A. Giessler and R. Platten, "Test Derivation from Net Models", in *Protocol Test Systems*, North-Holland, 1990, (Proc. IFIP TC6 2nd Workshop on Protocol Test Systems, Berlin, Germany, October 1989), pp. 77-91.
- [Bouricius 69] W.G. Bouricius, W.C. Carter and P.R. Schneider, "Reliability Modeling Techniques for Self-Repairing Computer Systems", *Proc. 24th ACM National Conference.*, 1969, pp. 295-309.
- [Camurati 88] P. Camurati and P. Prinetto, "Formal Verification of Hardware Correctness: Introduction and Survey of Current Research", *IEEE Computer*, July 1988, pp. 8-19.

- [Echtle 91] K. Echtle and Y. Chen, "Evaluation of Deterministic Fault Injection for Fault-Tolerant Protocol Testing", *Proc. 21st IEEE Int. Symp. Fault-Tolerant Computing*, Montréal, Québec, Canada, June 1991, pp. 418-425.
- [Choi 91] G.S. Choi, R.K. Iyer, R. Saleh and V. Carreno, "A Fault Behavior Model for an Avionic Microprocessor: A Case Study", in *Dependable Computing for Critical Applications*, Springer-Verlag, Vienna, Austria, 1991, pp. 177-195. (*Proc. 1st Int. Working Conference on Dependable Computing for Critical Applications*, Santa Barbara, CA, USA, August 1989)
- [Crouzet 82] Y. Crouzet and B. Decouty, "Measurements of Fault Detection Mechanisms Efficiency: Results", *Proc. 12th Int. Symposium Fault-Tolerant Computing*, Santa Monica, CA, USA, June 1982, pp. 373-376.
- [Czeck 90] E.W. Czeck and D.P. Siewiorek, "Effect of Transient Gate-Level Faults on Program Behavior", *Proc. 20th IEEE Int. Symposium Fault-Tolerant Computing*, Newcastle upon Tyne, England, June 1990, pp. 236-243.
- [Deo 74] N. Deo, *Graph Theory with Applications to Engineering and Computer Science*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1974.
- [Diaz 89] M. Diaz and C. Vissers, "SEDOS: Designing Open Distributed Systems", *IEEE Software*, November 1989, pp. 24-33.
- [Dugan 89] J.B. Dugan and K.S. Trivedi, "Coverage Modeling for Dependability Analysis of Fault-Tolerant Systems", *IEEE Transactions Computers*, vol. 38, June 1989, pp. 775-787.
- [Geist 90] R. Geist, M. Smotherman and R. Talley, "Modeling Recovery Time Distributions in Ultrareliable Fault-Tolerant Systems", *Proc. 20th IEEE Int. Symposium Fault-Tolerant Computing*, Newcastle upon Tyne, England, June 1990, pp. 499-504.
- [Goswami 90] K.K. Goswami et R.K. Iyer, "A Design Environment for Prediction and Evaluation of System Dependability", *Proc. 9th Digital Avionics Systems Conference*, Virginia Beach, VA, USA, October 1990, pp. 87-92.
- [Gunnello 89] U. Gunnello, J. Karlsson and J. Torin, "Evaluation of Error Detection Schemes Using Fault Injection by Heavy-ion Radiation", *Proc. 19th IEEE Int. Symposium Fault-Tolerant Computing*, Chicago, IL, USA, June 1989, pp. 340-347.
- [Joseph 88] M.K. Joseph and J. Bannister, "Coverage Estimation and Validation", *The Aerospace Corp. Report*, August 1988.
- [Laprie 85] J.-C. Laprie, "Dependable Computing and Fault-Tolerance: Concepts and Terminology", *Proc. 15th IEEE Int. Symposium Fault-Tolerant Computing*, Ann Arbor, MI, USA, June 1985, pp. 2-11.
- [Lyons 62] R.E. Lyons and W. Vanderkulk, "The Use of Triple-Modular Redundancy to Improve Computer Reliability", *IBM Journal*, April 1962, pp. 200-209.
- [Kime 86] C.R. Kime, "System Diagnosis", in *Fault-Tolerant Computing - Theory and Techniques* (D.K. Pradhan, ed.), Prentice-Hall, Englewood Cliffs, NJ, USA, 1986, pp. 577-632.
- [McGough 83] J. McGough, "Effects of Near-Coincident Faults in Multiprocessor Systems", *Proc. AIAA/IEEE Digital Avionics Systems Conference*, November 1983, pp. 16.6.1-16.6.7.
- [Powell 88] D. Powell, P. Verissimo, G. Bonn, F. Wacselynck and D. Seaton, "The Delta-4 Approach to Dependability in Open Distributed Computing Systems", *Proc. 18th IEEE Int. Symposium Fault-Tolerant Computing*, Tokyo, Japan, June 1988, pp. 246-251.
- [Raytheon 78] Raytheon Company, "Brassboard Fault-Tolerant Spaceborne Computer (BFTSC)", *Final Report*, December 1978.
- [Roy 70] B. Roy, *Subsets of Remarkable Vertices of a Graph*, Dunod, Paris, France, 1970. (In french).
- [Segall 88] Z. Segall, D. Vrsalovic, D. Siewiorek, D. Yaskin, J. Kownacki, J. Barton, D. Rancey, A. Robinson and T. Lin, "FIAT — Fault Injection Based Automated Testing Environment", *Proc. 18th Int. Symposium Fault-Tolerant Computing*, Tokyo, Japan, June 1988, pp. 102-107.
- [Siewiorek 82] D.P. Siewiorek and D. Johnson, "A Design Methodology for High Reliability Systems: the Intel 432[®]", in D.P. Siewiorek and R.S. Swarz, *The Theory and Practice of Reliable System Design*, Digital Press, Bedford, MA, USA, 1982, pp. 621-636.
- [Thévenod-Fosse 91] P. Thévenod-Fosse, "Software Validation by means of Statistical Testing: Retrospect and Future Direction", in *Dependable Computing for Critical Applications*, Springer-Verlag, Vienna, Austria, 1991, pp. 23-50. (*Proc. 1st Int. Working Conference on Dependable Computing for Critical Applications*, Santa Barbara, CA, USA, August 1989).
- [York 85] G. York, D.P. Siewiorek and Y.X. Zhu, "Compensating Faults in Triple Modular Redundancy", *Proc. 15th IEEE Int. Symposium Fault-Tolerant Computing*, Ann Arbor, MI, USA, June 1985, pp. 226-231.

RELIABILITY ANALYSES OF WORKSTATION FAILURE DATA

S. BROCKLEHURST*, K. KANOUN**, J.C. LAPRIE**, B. LITTLEWOOD*,
S. METGE**, P. MELLOR* and A. TANNER*

* The City University
Northampton Square
London EC1V OHB
UK

** LAAS-CNRS
7, Avenue du Colonel Roche
31077 Toulouse Cedex
FRANCE

SUMMARY

Experience of applying reliability models in the past has shown that the relative predictive performance of the models depends entirely on the context. It has been found that there is no one model that performs well over all data sets. It has also been found that for some data sets all models are in error. In such cases two techniques for improving predictive accuracy have been shown to be beneficial: i) recalibrating the raw model predictions and ii) using the results of trend tests preliminary to applying the models. These two techniques may be used separately or in combination. This paper is mainly devoted to the first technique but we will also show the benefit to be gained by the application of the second technique. We apply a number of reliability models to some failure data collected from a workstation installed at City University, together with the recalibration technique, and assess the performance of the resulting prediction systems.

1. INTRODUCTION

There are now many software reliability models available for the assessment and prediction of the failure behaviour of a program (or system) undergoing debugging or in operation. Some are universally bad, while the performance of others varies from data set to data set, but *it is not possible to select a globally good model* which will perform well over even a particular class of systems. This has led to the development of techniques which assess the predictive performance and compare the relative merits of these models in a particular context (ie. when applied to the data set under investigation). These techniques allow us to apply a number of models to the data of interest and select a "best" model in this context.

One of the techniques used also allows us to assess one aspect of the error in each of the models (called raw models hereafter), a kind of "bias", and to recalibrate the raw model predictions with respect to this error. This recalibrated prediction system is truly predictive and we can therefore use the above analysis techniques to assess the improvement gained via recalibration every time it is applied. It has proved to be beneficial (5) and the computational effort required for this technique is negligible compared with the effort required to apply many of the initial model prediction systems. It is recommended, therefore, that it is applied as a matter of course to all raw model prediction systems. The approach we suggest is that, for each data set, we should apply a number of raw models (preferably as many as is practical), and the recalibration technique to each of the raw models, and then use the analysis techniques to compare predictive accuracy of all the resulting prediction systems.

When the trend in the data exhibits *changes*, another technique consists of applying reliability growth models on the periods of trend between such changes. Application of

trend tests to the collected data allows these reliability growth, reliability decay and stable periods to be identified, and thus the data to be partitioned into subsections. Reliability growth models may then be applied to each subsection displaying trend in accordance with the model assumptions, thus improving the accuracy of the results (12), (13), (2).

One of the major barriers facing research in the area of software reliability modelling is the lack of the data required in order to make predictions about the reliability of a program or system. Collecting the appropriate data can be a costly process and testing has to be conducted under a suitable operational profile (ie. a user profile) in order to get accurate reliability predictions. In this paper we shall analyse a new set of data which shows the failure behaviour of a single-user work station, installed at the City University in March 1985 and subsequently used over a period of four years.

2. RAW RELIABILITY MODELS

The data we are considering are times, t_1, t_2, t_3, \dots (which can be C.P.U. execution time, calendar time, or any other applicable measure) between successive failures resulting from first occurrence of unique faults of a system. In the case of perfect debugging this will correspond to the times between successive failures arising from different faults. For the data presented in this paper corrections were (generally) not carried out; only the first failure arising from each fault was extracted in order to obtain the required inter-failure times. This effectively simulates what *would* happen under perfect debugging.

For simplicity we shall be considering one-step-ahead predictions. Using the previous inter-failure times, t_1, t_2, \dots, t_{j-1} the raw models provide a prediction of the current (and as yet unobserved) inter-failure time, T_j , in the form of a *predictive cumulative distribution function (cdf)*,

$$\dots \hat{F}_j(t) = \hat{P}_r (T_j \leq t) \quad \dots(2.1)$$

From (2.1) we also have a *predictive probability density function (pdf)* for T_j ,

$$\hat{f}_j(t) = \hat{F}'_j(t) \quad \dots(2.2)$$

These can be thought of as estimates of the true underlying *cdf* and *pdf*, $F_j(t)$ and $f_j(t)$, respectively.

The models considered in this paper are *parametric* models; that is, they assume a form for the *pdf (cdf)* which depends on some unknown parameter(s). Estimates for these parameters are made at each stage, j , by using the *previous* data and the method of maximum likelihood. These parameters are then substituted into the *pdf (cdf)* in order to make predictions about T_j . Note that performance of these models will depend not only on their precise mathematical structure, but on the maximum likelihood inference technique and the substitution rule for prediction. It should be noted that these last two approaches to statistical inference and prediction are chosen here for convenience: other techniques, such as Bayesian inference, tend to be computationally intensive.

The models which are applied in this paper are the Duane (*D*) (10), (11), Hyperexponential (*HE*) (18), (19), Keiller Littlewood (*KL*) (15), (16), Littlewood (*LM*), (20) Littlewood non-homogeneous Poisson process (*LNHPP*) (1), Littlewood Verrall (*LV*) (21) and Musa Okumoto (*MO*) (24) models. Since most of these models are well known the details are omitted from this paper.

Many of the parametric models have the property that they tend towards simpler models as their parameters (or functions of their parameters) tend to extreme values. In particular the *HE*, *MO*, *LM* and *LNHPP* models tend to the stable reliability homogeneous Poisson process (*HPP*) when they are applied over regions of data which do not exhibit growth. There are other ways in which these models can tend to simpler solutions, details of which are summarised in (6) .

All the models considered in this paper can model stable reliability or reliability growth (ie. constant failure rate or monotonically decreasing failure rate) while the *DU*, *LV* and *KL* models can also model reliability decay (ie. monotonically increasing failure rate). None of the models as they are applied in this paper, however, are able to model trend changes (for example the transition from stable reliability to reliability growth), since they all assume a smooth trend. However, even though each fitting of the model will be unable to represent a change in the trend of the data, a sequence of predictions from the model may respond to such a change. That is, when applying a model over a succession of one-step-ahead predictions upon a data set containing changes in trend, it may be that the series of predictions *themselves* do not exhibit smooth trend.

3. ANALYSIS OF PREDICTIVE QUALITY AND RECALIBRATION

In this section the methods for assessing and comparing the performance of the various models are briefly described. They all depend on a comparison between the estimated *cdf* or *pdf* (see (2.1) and (2.2)) at stage *j*, and the (later observed) realisation, *t_j*, of the next inter-failure time, *T_j*. Suppose we have *q* inter-failure times altogether. Then we can apply the raw models summarized in section 2, to the data *t₁, ..., t_{j-1}*, to obtain our prediction for *T_j*, for *j = s, ..., q*, say, where *s* is a suitably large number. We then have what we shall refer to as our "prediction system" for each of the models, ie.

$$\{ \hat{F}_j(t), \hat{f}_j(t) ; j = s, \dots, q \} \tag{3.1}$$

The *u*-plot

Our first technique involves substituting the (later observed) *t_j* into the (earlier computed) predictive *cdf*:

$$u_j = \hat{F}_j(t_j) \qquad j = s, \dots, q \tag{3.2}$$

If our prediction system is identical to the truth, ie. $\hat{F}_j \equiv F_j \forall j$, then the *u*'s should behave as if they come from a *U(0,1)* distribution. The *u*-plot is the sample *cdf* of the *u*'s defined in (3.2) (1). Departures of the *u*-plot from the 45° line, which is the *U(0,1)* *cdf*, indicate that our prediction system is inaccurate in some way. Informally, the *u*-plot is a powerful means of detecting various kinds of *consistent bias* in predictions, ie. situations where the prediction errors are in some sense *stationary*.

The *y*-plot

Let

$$x_j = \log(1-u_j) \qquad j = s, \dots, q \tag{3.3}$$

and

$$y_r = \frac{\sum_{j=s}^r x_j}{\sum_{j=s}^q x_j} \quad r = s, \dots, q \quad \dots(3.4)$$

Then the y -plot is constructed similarly to the u -plot by drawing the sample *cdf* of the y 's. Note, from (3.3) and (3.4), that the y -plot preserves the order of occurrence of the u 's over time whereas the u -plot does not. If our prediction system has captured the trend (eg. reliability growth) in the data then the y -plot should be close to the 45° line. Thus the y -plot is a means of detecting those cases where the prediction errors are *non-stationary*.

For a more detailed explanation of the u - and y -plots see (1), (9), (15) and (16).

The prequential likelihood ratio

It should be noted that it is possible for a prediction system to give good u - and y -plots and yet still be inaccurate; for example it could be very *noisy*, so that individual predictions emanating from it are very inaccurate even though on average there is no bias, and there is no evidence of non-stationarity in the errors of prediction. For this reason we use a further measure called the prequential likelihood ratio (*PLR*) which is intended as global comparison of goodness of prediction for one prediction system versus another.

Suppose we have two prediction systems, a and b , say. Then the *PLR* is defined to be

$$PLR = \frac{\prod_{j=s}^q \hat{f}_j^\alpha(t_j)}{\prod_{j=s}^q \hat{f}_j^\beta(t_j)} \quad \dots(3.5)$$

Notice that, unlike the u - and y -plots, this measure depends upon the *pdfs* rather than the *cdfs*.

If $PLR \rightarrow \infty$ as $q \rightarrow \infty$ then we would choose model α as being the better of the two models. Conversely if $PLR \rightarrow 0$ as $q \rightarrow \infty$, we would favour model β over model α . As we clearly never have $q \rightarrow \infty$ in practice, the best we can do is to look for steady increases and decreases in our *PLR* plots of one model versus another over the whole data set. In the analysis that follows we actually plot the $\log(PLR)$ at stage j against j . To see a more detailed explanation of the use of the *PLR* in the context of software reliability modelling see (1).

The recalibration technique

If we have captured the trend in the data, ie. the prediction errors are stationary, but the predictions are biased then the (approximately) consistent departure of the prediction system from the truth is represented by the departure of the u -plot from the 45° line. This is where the *recalibration technique* can use the u -plot to eliminate this bias from the raw prediction system and thus construct a new improved prediction system.

This technique consists of using the joined up u -plot, G_i , based on *previous predictions* of t_s, \dots, t_{i-1} , in order to adjust the current raw prediction,

$$\hat{F}_i(t) = G_i(\hat{F}_i(t)) \quad \dots(3.6)$$

For more details of this technique see (3), (5) and (17). We can repeat this recalibration procedure over $i = p, \dots, q$, where p -s is suitably large and we then have a recalibrated prediction system,

$$\{\hat{F}_i(t), \hat{f}_i(t) ; i = p, \dots, q\} \quad \dots(3.7)$$

We can then use the u - and y -plots and the *PLR* as outlined above in exactly the same manner as with the raw prediction system to assess our new recalibrated prediction system.

Since we are using the *PLR* to compare our prediction systems we will use the spline recalibration technique (see (7) and (8)) which consists of smoothing each G_i using least squares cubic splines before recalibration. This smoothing is necessary since discontinuity in the derivative of G_i causes the *PLR* to report badly about the resulting recalibrated predictions, although most predictions of interest will be altered little by smoothing (3).

All the raw prediction systems from the reliability models referred to in section 2 are spline-recalibrated. An *S* added onto the end of the model names will be used to denote the recalibrated prediction system (eg. *HES*, *MOS*, *LVS* etc.). This results in a number of different prediction systems for each data set, the performance of which we can compare using techniques for the analysis of predictive accuracy described above.

Preliminary data processing

A global method incorporating preliminary data processing has been investigated in (13) and (14). Such preliminary data processing makes it possible to identify outliers (doubtful data) and to test for changes in the trend in order that the reliability growth models can be applied to appropriate subsections of the data. Here we shall consider only this latter example of preliminary processing.

Many software reliability models assume that the inter-failure time data exhibits reliability growth. It is certainly the case that if we are truly debugging our system we would, on average, expect to see reliability growth. However, it has been observed that, particularly early on in the operational life of a system, growth may not be present in spite of attempted bug-fixing, or even that there is reliability decay. One explanation of this phenomenon is that, early on in the life of a system, we are continually exploiting new parts of the system. Since we are not frequently reusing the same parts of the system, we might not expect to see significant growth, even though fixes are taking place. Other reasons include fault dependency, varying delays between identification and removal of faults, and even small specification changes.

As mentioned in section 2, some of the models considered in this paper can handle reliability decay as well as stable reliability or reliability growth. Change points in the trend in the data, however, for example from stable reliability to reliability growth, may present difficulties for all these models and result in inaccurate predictions. It is therefore important to be able to identify those subsections of the data within which there are no trend-change points. One method of determining whether such a subsection of data

really does exhibit reliability growth (or decay, or stability) uses the Laplace statistic. For inter-failure time data t_1, t_2, \dots, t_n , this is

$$L(t_1, t_2, \dots, t_n) = \frac{\sum_{j=1}^{n-1} \tau_j}{(n-1) \frac{\tau_n}{2}} - \frac{\tau_n}{2} \dots (3.8)$$

where $\tau_j = \sum_{r=1}^j t_r$

Large negative values of this suggest there is reliability growth, large positive values decay. Since the statistic is approximately normally distributed (9), under the null hypothesis that the data come from a (trend-free) homogeneous Poisson process, very simple tests can be conducted for growth (or decay) in a particular data vector. More informally, it is often instructive to plot the changes in the statistic as the data vector increases in size, as we shall see later.

4. THE DATA SET

The data used in this paper comes from operational use of a single-user work station which was installed at the City University on the 18th March, 1985.

Data collected included real (or wall-clock) time of occurrence of each failure (recorded to the nearest minute), together with the identity of the particular fault which caused the failure (so that time to first occurrence of each unique fault can be recovered and hence the required inter-failure times). Additionally details of the type of usage and the version of the operating system in use at the time of each failure, the severity of failure, and type of the associated fault were recorded.

In order to assess software reliability by statistical analysis of time of occurrence of first failure due to each fault, it is necessary to have a measure of the amount of use to which the software has been subjected (22). Real, or wall-clock, time is rarely appropriate. In this case it was decided that "hands-on" time at the work station was suitable. This was because the failures were recorded as a result of observation by the user, and because many of them were "usability problems", i.e., the encountering of features of the system which caused difficulty for the user, even though the system was behaving according to specification.

For reasons of space we shall analyse here only two subsets of this data. The largest, *USBAR*, consists of times between successive failures of certain unique faults (i.e. ignoring failures of previously seen faults). Almost all failures are included here, including usability problems: we omit only those occurring during power-on, power-off and machine idle time (i.e. machine on but not being used). This data comprises 397 inter-failure times in all. From this data set we have extracted a subset, *TSW*, comprising only *software* failures; there are 129 of these. A more comprehensive analysis of this data, involving other subsets, can be found in (6).

760.	758.	303.	6.	22.	14.	42.	4.	84.	15.	221.	14.	15.	41.	1.	153.
409.	54.	24.	44.	180.	397.	19.	145.	36.	54.	1337.	163.	8.	1.	17.	16.
87.	19.	29.	0.5	300.	360.	10.	11.	100.	252.	460.	179.	3.	24.	253.	163.
54.	137.	328.	3.	9.	12.	18.	9.	75.	15.	366.	428.	212.	115.	264.	269.
276.	1.	999.	30.	495.	472.	344.	550.	131.	47.	92.	863.	991.	35.	9549.	249.
607.	83.	614.	352.	673.	4179.	111.	75.	407.	288.	894.	1314.	845.	55.	409.	36.
15.	1960.	60.	19.	20.	79.	24.	1737.	7984.	10.	20.	338.	250.	1682.	212.	287.
56.	4973.	3500.	59.	98.	2439.	1812.	6203.	385.	3500.	4892.	687.	62.	2796.	3268.	3845.
76.															

Table 1: Data set *TSW* (total number=129)

Table 1 shows the raw data for *TSW*. Space restrictions preclude our showing a similar table for *USBAR*, but Fig 1 shows this plotted as a cumulative number of failures against total elapsed time. It is clear that in each case there is *overall* reliability growth. The times in the later part of the table tend to be bigger on average than earlier ones; the slope of the plot in the figure (representing the rate of occurrence of failures) shows an overall decrease.

However, within this overall pattern of growth, there is some variation. This is clear even from the raw data of *USBAR* plotted in Fig 1. A more detailed analysis via the Laplace statistic is shown in Figs 2 and 3. In Fig 2 the Laplace statistic starts to show a consistent trend downwards about halfway through the data set; in Fig 3, there seems to be reliability growth after only the first few observations.

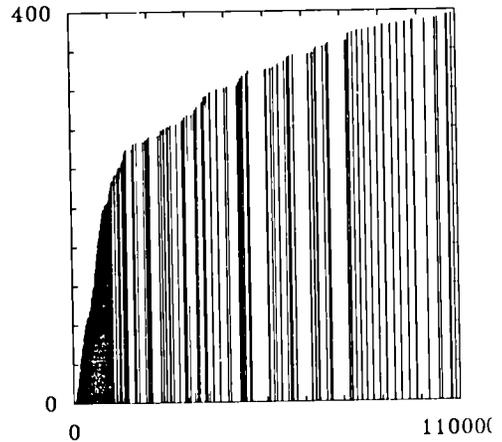


Fig. 1: Cumulative number of failures against total time for data set *USBAR*

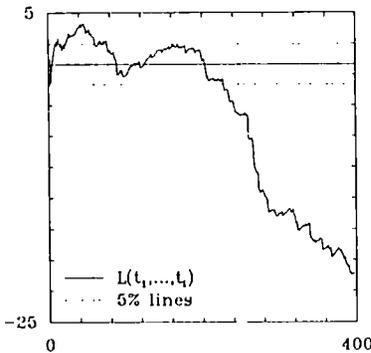


Fig. 2: Laplace statistic against failure number for data set *USBAR*

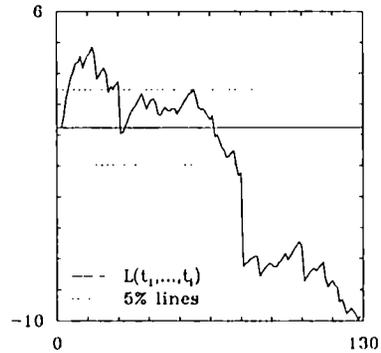


Fig. 3: Laplace statistic against failure number for data set *TSW*.

5. DATA ANALYSIS USING SEVERAL PREDICTION SYSTEMS

Due to space limitations and since we are mainly interested in model recalibration in this paper, all the models referred to in section 2 are first applied blindly (ignoring the Laplace statistic analysis on the various data sets) over the *whole* of the data available. However, in order to show the impact of taking into account this analysis, we shall apply one of the models, *HE*, over a period of time which takes account of the fact that in each of the data sets reliability growth does not start at the beginning of the data set. We shall use *HEc* to denote this additional prediction system: the model is applied over the range, t_{c+1}, t_{c+2}, \dots . That is t_{c+1}, \dots, t_{c+s} are used to make predictions about T_{c+s+1} , then $t_{c+1}, \dots, t_{c+s+1}$ are used to make predictions about T_{c+s+2} , and so on. An informal choice of $c = 144$ for *USBAR* and $c = 14$ for *TSW* was made from the preliminary trend analysis.

In this section we shall discuss application of the 7 parametric models referred to in section 2 (including the second application of the *HE* model, *HEc*), each both raw and spline-recalibrated, to data sets *USBAR* and *TSW*. The quality of the 16 resulting prediction systems, with respect to one-step-ahead predictions, will be discussed for each data set based on the *u*-plot, the *y*-plot and the prequential analysis techniques described in section 3.

In the application of the raw parametric models we have to decide on the number of inter-failure times which will be used in order to get the first prediction. If this number is too small then early predictions will be likely to be too noisy. On a purely informal basis we have chosen to use the first 20 inter-failure times to get the first raw prediction. We also have to decide at what point to begin recalibrating our raw predictions. If we recalibrate too early then the *u*-plot for recalibration will contain too few points and our estimate of the *cdf* of the current *u*, even in the presence of stationarity in the raw model errors, may be inaccurate, and hence recalibration may not be very efficient. We have, again informally, chosen to use the first 15 raw predictions in the *u*-plot to achieve the first recalibrated prediction. Thus, in the notation of section 3, $s = 21$ and $p = 36$. In the case of the second application of the *HE* model (ie. *HEc*) the resulting predictions (raw and recalibrated) will start later on in the data than those from all the other prediction systems.

Table 2 shows the significance levels (see (23)) for the *Kolmogorov-Smirnov* distances of the *u*- and *y*-plots from the 45 line for the various prediction systems on the data sets. The *Kolmogorov-Smirnov (K-S)* distance is the maximum vertical distance of the plot from the 45° line. Here we show in lower case the levels for the range of the *HEc* model predictions, ie. $T_{c+37}, T_{c+38}, \dots$, whilst the upper case refers to the *u*- and *y*-plots which include predictions of T_{36}, T_{37}, \dots . The *u*- and *y*-plots for the shorter range of predictions are included since we wish to compare these models with *HEc*. Raw predictions prior to the stage of the first recalibrated predictions (ie. the first 15 raw predictions) are not in-

	<i>USBAR</i>				<i>TSW</i>			
	<i>c = 144</i>				<i>c = 14</i>			
	<i>RAW</i>		<i>S</i>		<i>RAW</i>		<i>S</i>	
	<i>y</i>	<i>u</i>	<i>y</i>	<i>u</i>	<i>y</i>	<i>u</i>	<i>y</i>	<i>u</i>
<i>HE</i>	F	F	Fc	F	Db	F	A	Cd
<i>HEc</i>	f	f	b	f	b	f	a	d
<i>MO</i>	F	F	Df	Df	Bd	F	Df	A
<i>DU</i>	Fa	F	Fc	Df	A	F	Ac	A
<i>LM</i>	F	F	F	Dc	F	F	F	A
<i>LNHPP</i>	F	F	F	Dc	F	F	F	A
<i>LV</i>	Fa	F	Fa	Cf	A	F	A	A
<i>KL</i>	Ea	F	Cb	Af	A	F	A	A

A: insignificant at 20% B: 10-20% C: 5-10%
 D: 2-5% E: 1-2% F: significant at 1%

The upper case letters represent the significance levels for $p = 36$ while the lower case letters represent the significance levels with $p = c + 37$ (if they differ).

Table 2: *u*- and *y*-plot significance levels for predictions of T_p, T_{p+1}, \dots .

cluded in the plots since we want to compare recalibrated and raw predictions over the same range of data.

Data set USBAR

From table 2 it can be seen that both the u - and y -plots for the various raw models on this data set are highly significant with the exception of the y -plots for the DU , LV and KL models over the latter half of the data set, where the y -plots are insignificant. The median predictions in Fig 4 show great disagreement between the models at the end of the data set.

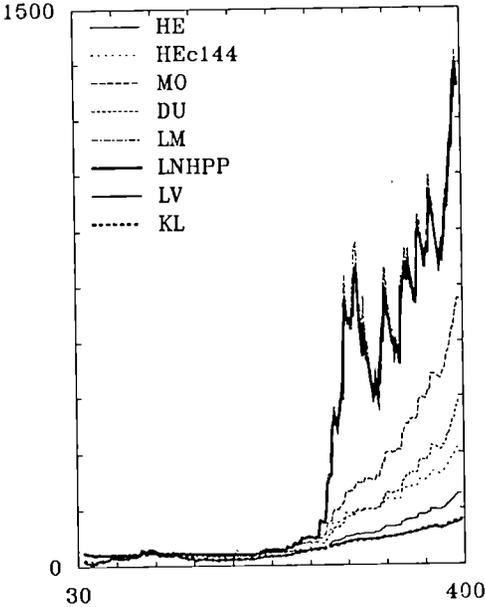


Fig. 4: Median predictions from the raw models for data set USBAR.

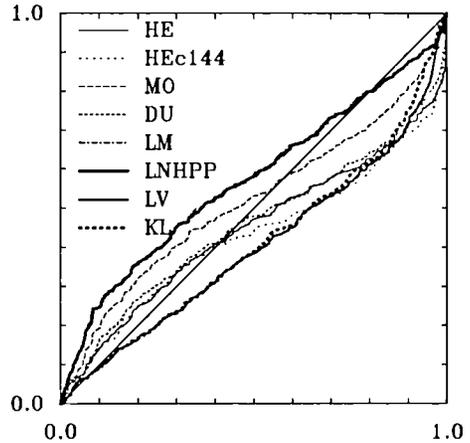


Fig. 5: u -plots from the raw models for data set USBAR for predictions of T_{181} , T_{182} , .. T_{397} for HEc and T_{36} , T_{37} , .. T_{397} for the remaining models.

The corresponding u -plots for these models (see Fig 5) indicate that the LM and $LNHP$ models are generally optimistic, while the LV and KL models are generally pessimistic. For the other models the bias in the distributions would appear to be more complicated than simple optimism and pessimism. Over most of the data prior to the 200th failure the HE , MO , LM and $LNHP$ models tend to HPP predictions (notice, from Fig 2, that this coincides with the region where the Laplace statistic indicates absence of global reliability growth).

In the case of HE and HEc , the y -plot compared over the latter half of the data gives K - S distances of 0.152 and 0.124,

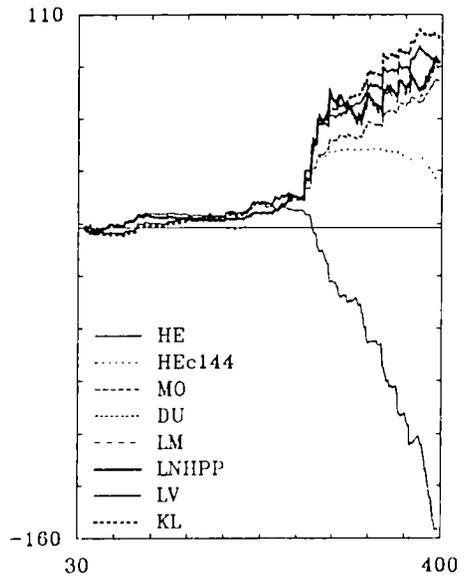


Fig. 6: $\text{Log}(PLR)$ for the raw models versus DU for data set USBAR.

respectively, indicating that not considering the earlier data may indeed have enabled this model to better capture the trend in the later failure data. Indeed, Fig 6, which shows the $\log(PLR)$ plot for all these raw models against the *DU* model, as the data evolves, shows how dramatic this improvement is from $i \approx 270$. This point may correspond to a second change point (see change of slope of Laplace statistic plot in Fig 2) and from this point the *HE* model experiences great difficulties, whereas *HEc* does not. This plot also suggests that beyond this point the *MO*, *LM*, *LNHPP*, *LV* and *KL* models may be the better predictors, but it is clear, from the *u*-plots, that all these raw models are in error.

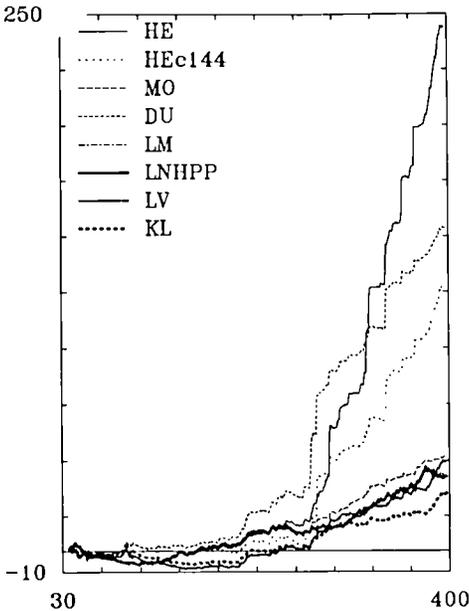


Fig. 7: $\log(PLR)$ for the recalibrated versus raw prediction systems for data set *USBAR*.

Fig 7, which shows the $\log(PLR)$ plot of the recalibrated predictions versus the raw predictions, indicates that there is a huge improvement to be gained via recalibration in the latter part of the data set for those raw models which were initially performing particularly badly (ie. *HE* and *DU*) and for *HEc* there is also dramatic improvement, while for the remaining models there is slight improvement. Comparison of figures 5 and 8 show that there is dramatic improvement

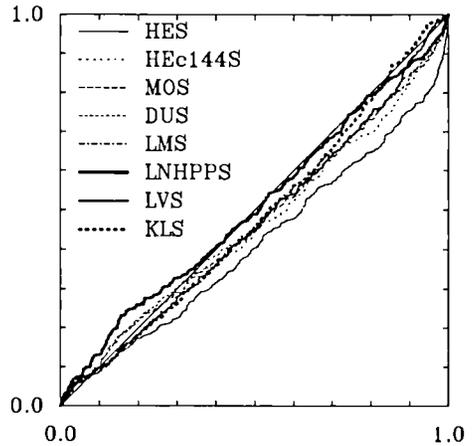


Fig. 8: *u*-plots from the spline-recalibrated models for data set *USBAR* for predictions of $T_{181}, T_{182}, \dots, T_{397}$ for *HEcS* and $T_{36}, T_{37}, \dots, T_{397}$ for the remaining models.

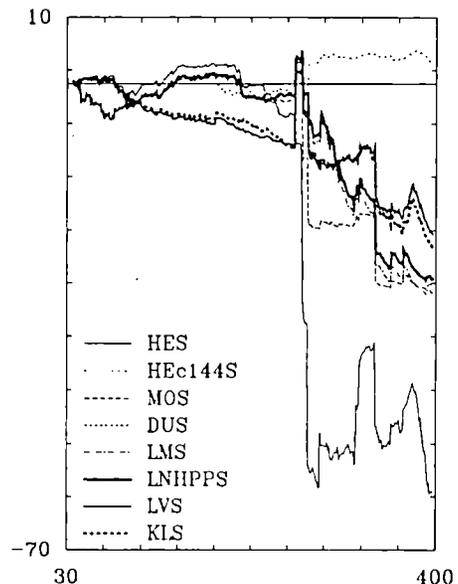


Fig. 9: $\log(PLR)$ for the spline-recalibrated models versus *DUS* for data set *USBAR*.

in the u -plots via recalibration, although in the case of the HE model it would seem that there is still pessimism in the recalibrated predictions. It is interesting to observe (see Fig 9¹) that the recalibrated models which are now performing the best, according to the PLR , do not come from those raw models which were initially performing the best. In particular, the DU model, before recalibration, was steadily worse than the others (with the exception of HE), while after recalibration it is one of the better predictors together with HEC . Fig 9 also shows the benefit from applying the HE model (HEc) on data from 145; the improvement in the recalibrated HEc predictions over the recalibrated HE predictions indicates that the raw HEc model has indeed better captured the trend in the data. Comparison of figures 6 and 9 show that, in general, recalibration has had the effect of bringing the models into much closer agreement and the recalibrated median predictions in Fig 10 are indeed in closer agreement than the raw medians. Comparison of figures 4 and 10 shows that some of the model medians have been adjusted for optimism, while others have been adjusted for pessimism. Although these are closer in agreement than the raw predictions they still diverge as the data evolves.

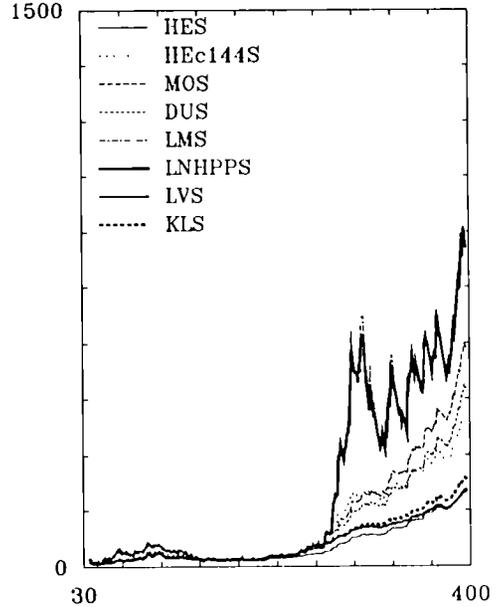


Fig. 10: Median predictions from the spline-recalibrated models for data set $USBAR$.

Data set TSW

For this data set it can be seen from Fig 11 that the medians again diverge as the data evolves. Prior to the point at which these predictions diverge, the medians from some of the raw models appear to be coincident. This is because up to $i \approx 66$ the HE , MO , LM and $LNHP$ generally tend to an HPP ; comparing this with Fig 3 reveals that this again coincides with absence of global reliability growth. From table 2 it can be seen that all the u -plots for the raw models are again highly significant and Fig 12 suggests that the LM and $LNHP$ median predictions are optimistic while the LV and KL median predictions are pessimistic. The $\log(PLR)$ plot in Fig 13 shows that there is little to choose between the various raw prediction systems over much of the data set with the exception of the large jump at $i = 79$ ².

- 1 The large jumps downwards at $i = 267$ and $i = 273$, for HES , coincide with unusually large inter-failure times, t_{267} and t_{273} , and each HES predictive distribution clearly has a smaller right hand tail compared with the other recalibrated models. Comparison with Figs 6 and 7 suggest that this problem was present in the raw model predictions and is not an artefact of recalibration. Fig 7 suggests that this problem was also present for the raw DU model, but that it was eliminated by recalibration.
- 2 The large jump upwards in this plot at $i = 79$ is again due to the occurrence of a particularly large inter-failure time ($t_{79} = 9549$). The differences in performance here for the different models are also present in the recalibrated predictions and so in this case recalibration has done little to help

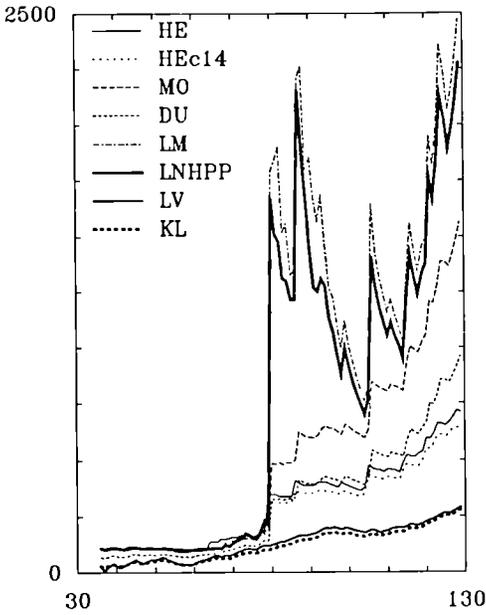


Fig. 11: Median predictions from the raw models for data set TSW.

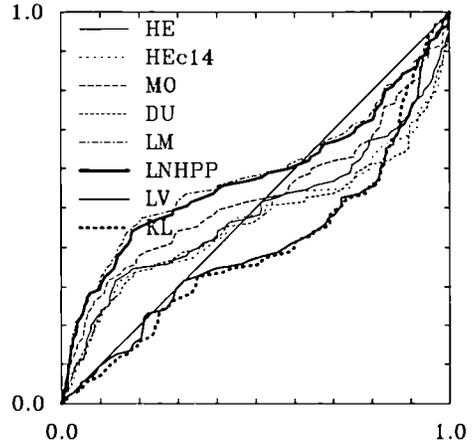


Fig. 12: u -plots from the raw models for data set TSW for predictions of $T_{51}, T_{52}, \dots, T_{129}$ for HEC and $T_{36}, T_{37}, \dots, T_{129}$ for the remaining models.

For this data set we are again interested in the recalibrated predictions since the raw predictions are clearly in error. After recalibration we can see, from table 2 and Fig 14 that there is improvement in all the u -plots and most of them have become insignificant, while the recalibrated medians shown in Fig 15 are now in much closer agreement. It is surprising, at first, that the HE and HEC median predictions are so close to the others while the u -plot significance levels are so different from those of the other models. In fact,

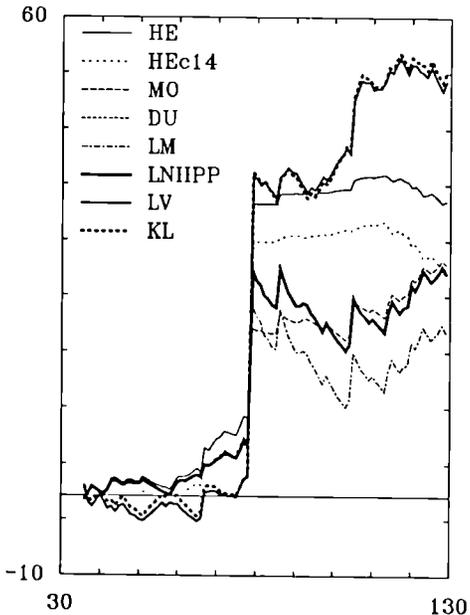


Fig. 13: $\text{Log}(PLR)$ for the raw models versus DU for data set TSW.

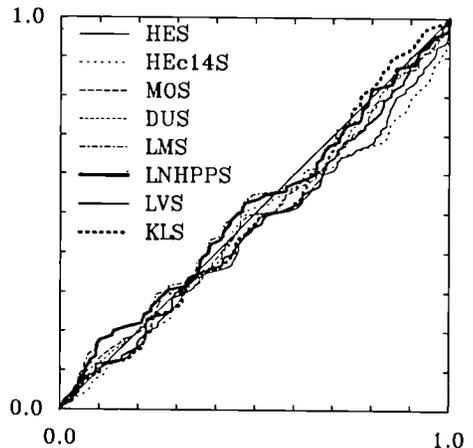


Fig. 14: u -plots from the spline-recalibrated models for data set TSW for predictions of $T_{51}, T_{52}, \dots, T_{129}$ for HECs and $T_{36}, T_{37}, \dots, T_{129}$ for the remaining models.

this is due to the K - S distances for these models being a result of pessimistic recalibrated predictions for *large* inter-failure times (ie. in the upper part of the distribution), while the u -plot near 0.5 is quite close to the 45° line (see Fig 14).

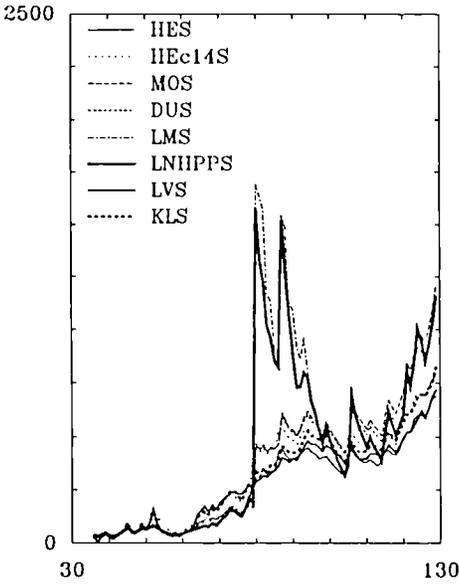


Fig. 15: Median predictions from the spline-recalibrated models for data set TSW.

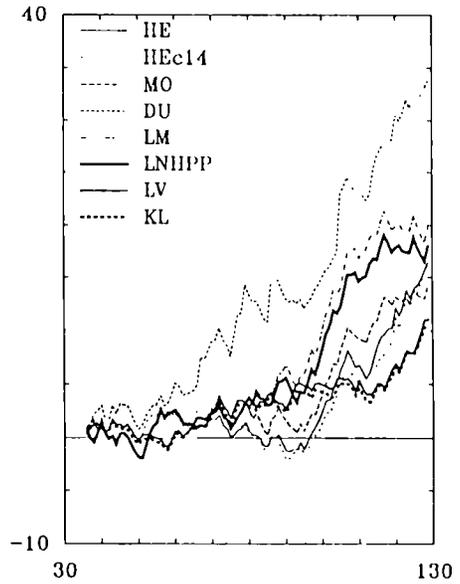


Fig. 16: $\text{Log}(PLR)$ for the recalibrated versus raw prediction systems for data set TSW.

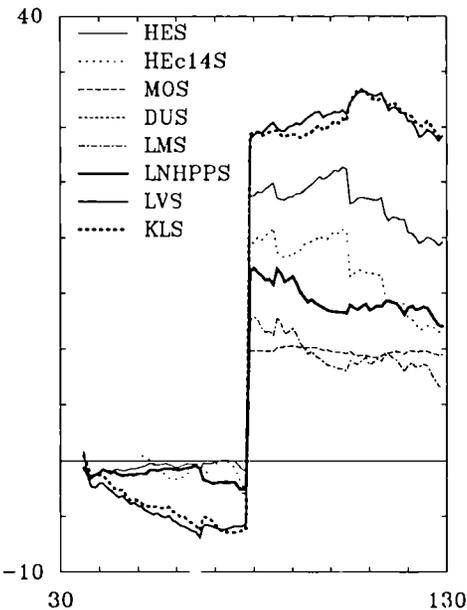


Fig. 17: $\text{Log}(PLR)$ for the spline-recalibrated models versus DUS for data set TSW.

Fig 16 shows the improvement gained by recalibration, for the various models, according to the $\log(PLR)$; for the DU model, in particular, there seems to be a consistent improvement over the whole data set and for the other models there is improvement later on in the data while earlier on there is little to choose between the recalibrated and raw predictions. The $\log(PLR)$ plot in Fig 17 shows that, after recalibration, there is little to choose between the various prediction systems over much of the data set with the exception of the large jump at $i = 79$.

6. GENERAL COMMENTS

For the two data sets selected it is clear that recalibration is effective in improving on the raw model predictions, particularly later on in the data. It also seems that raw models which are initially comparatively bad, tend to be vastly improved by this technique so that, after recalibration, they are comparable with (and sometimes better than) the recalibrated versions of models which were initially better. This means that it may not always be necessary to apply a sophisticated model, when a cruder model, with recalibration, will do better. As stated earlier, the only requirement we wish in the initial raw model, is that its predictions (as opposed to how well it fits the data in retrospect) have captured the trend in the data. In this situation we can expect recalibration to be efficient. These results support those reported earlier in (5).

There is also evidence that the comparative performance of the prediction systems may change not only for different data sets, but for different ranges of predictions within one data set. This is particularly evident for the raw model predictions where performance for *USBAR* diverges at the end of the data while earlier in the data the various raw model predictions are closer in agreement. After recalibration, although performance of the resulting prediction systems is in closer agreement, there is still evidence that relative performance is different over different intervals of data.

For each of the data sets, there appears to be a change point before which the models are comparable in their performance, and after which their performance diverges. These points also coincide with the stage at which recalibration starts to give steady (and often dramatic) improvement for most of the models. It is clear that, for these data sets, the change point in the data creates difficulties for some of the raw models; similar situations have also been noticed in (12). It also seems likely that such change points may affect the efficiency of recalibration, since it may cause non-stationarity in the departure of the raw model predictions from the truth. For data set *USBAR*, for example, although the *u*-plots (and the predictions) are vastly improved by recalibration it can be seen that they are mostly still poor. In fact the significance levels vary depending on which range of data is included in the plots (see table 2). This indicates non-stationarity in the error in the recalibrated models which is likely to be the result of change points in the data. In such situations recalibration may be expected to improve the predictions, but still leave room for further improvement.

There are alternative ways that might further improve predictions. In (4; 6) the issues of applying raw reliability models and the recalibration technique using *moving windows* of fixed sizes across the data and the raw model predictions, respectively, are investigated. Alternatively the raw models and/or the recalibration technique could be applied using trend test results, in a similar manner to *HEc*. The latter technique is likely to be more efficient since using fixed moving windows may cause data to be thrown away unnecessarily, resulting in less bias in the predictions but at the expense of increased noise. For raw model application we wish to find windows of data which have stationary trend in order to optimise resulting predictive accuracy, whereas for recalibration we wish to find windows of raw model predictions for which the *prediction errors* are stationary.

For simplicity, in this paper, the analysis of the prediction systems resulting from the various models and from the recalibration technique, is based upon one-step ahead predictions. It is clear that there are many other types of predictions, further into the future, which may be of interest to the user and that the prediction system which gives the best one-step ahead predictions will not necessarily give the best predictions further into the future. This means that if we wish to use the techniques described in section

3, in order to assess the performance of such predictions, we ideally need to use predictions of the same *type* as the prediction of interest. It follows then, that we also need to use the same *type* of prediction in the *u*-plot for recalibration of the prediction of interest. On a single data set, it is not clear how to extend these techniques to predictions which are far into the future, simply because we are unlikely to see enough (if any) realisations of such predictions. Issues of how we achieve such predictions for the raw models, how we assess the accuracy of such predictions, and how we might recalibrate such predictions, need to be addressed in future work.

REFERENCES

- (1) ABDEL GHALY, A.A., CHAN, P.Y., and LITTLEWOOD, B. (1986). "Evaluation of Competing Software Reliability Predictions", *IEEE Trans. on Software Engineering*, vol. SE-12, no.9, pp. 950-967.
- (2) BASTOS MARTINI, M.R., KANOUN, K. and MOREIRA DE SOUZA, J. (1990). "Software Reliability Evaluation of the TROPICO-R Switching System", *IEEE Trans. on Reliability*, vol. 39, no. 3, pp. 369-379.
- (3) BROCKLEHURST, S. (1987). "On the Effectiveness of Adaptive Software Reliability Modelling", C.S.R. Technical Report.
- (4) BROCKLEHURST, S. (1989) "A Non-Parametric Approach to Software Reliability Modelling", C.S.R. Technical Report.
- (5) BROCKLEHURST, S., CHAN, P.Y., LITTLEWOOD, B. and SNELL, J. (1990). "Recalibrating Software Reliability Models", *IEEE Trans. on Software Engineering*, vol. SE-16, no. 4, pp. 458-470.
- (6) BROCKLEHURST, S., MELLOR, P. and TANNER, A. (1991). "A Multi-Modelling Approach to Software Reliability Prediction", C.S.R. Technical Report, in preparation.
- (7) CHAN, P.Y. (1986). "Software Reliability Prediction", Ph.D. dissertation, City University.
- (8) CHAN, P.Y., LITTLEWOOD, B. and SNELL, J. (1985). "Parametric Spline Approach to Adaptive Reliability Modelling", C.S.R. Technical Report.
- (9) COX, D.R. and LEWIS, P.A.W. (1966). *Statistical Analysis of Series of Events*, Methuen, London.
- (10) CROW, L.H. (1977). "Confidence Interval Procedures for Reliability Growth Analysis", U.S. Army Material Syst. Anal. Activity, Aberdeen, MD, Tech. Report 197.
- (11) DUANE, J.T. (1964). "Learning Curve Approach to Reliability Monitoring", *IEEE Trans. Aerospace*, vol. 2, pp. 563-566.
- (12) KANOUN, K. and SABOURIN, T. (1987). "Software Dependability of a Telephone Switching System", *Proc. 17th IEEE Int. Symp. on Fault-Tolerant Computing (FTCS- 17)*, pp. 236-241, Pittsburgh, Pennsylvania, 68 July.
- (13) KANOUN, K., LAPRIE, J.C. and SABOURIN, T. (1988). "A Method for Software Reliability Growth Analysis and Assessment", *Proc. of Int. Workshop on Software Engineering and its Applications*, pp. 859-878, Toulouse, France, 5-9 Dec.
- (14) KANOUN, K. (1989) "Software Dependability Growth characterization, modeling and evaluation", Doctorat ès-Sciences thesis, Institut National polytechnique de Toulouse, LAAS report n 89320, Sept.; in French.
- (15) KEILLER, P.A., LITTLEWOOD, B., MILLER, D.R. and SOFER, A. (1983). "On the Quality of Software Reliability Predictions", *Proc. of NATO ASI on Electronic Systems Effectiveness and Life Cycle Costing*, (ed. J. Skwirzinski), pp. 441-460, Springer-Verlag, Heidelberg, Germany.

- (16) KEILLER, P.A., LITTLEWOOD, B., MILLER, D.R. and SOFER, A. (1983). "Comparison of Software Reliability Predictions", *Proc. 13th IEEE Int. Symp. on Fault-Tolerant Computing (FTCS-13)*, pp. 128-134.
- (17) KEILLER, P.A. and LITTLEWOOD, B. (1984). "Adaptive Software Reliability Modelling", *Proc. 14th IEEE Int. Symp. on Fault-Tolerant Computing (FTCS-14)*, pp. 108-113.
- (18) LAPRIE, J.C. (1984). "Dependability Modelling and Evaluation of Software and Hardware Systems", *Invited survey to the 2nd GI/NTG/GMR Conference on Fault-tolerant Computing*, Bonn, Germany, September 19-21.
- (19) LAPRIE, J.C., KANOUN, K., BEOUNES, C. and KAANICHE, M. (1991). "The KAT (Knowledge-Action-Transformation) Approach to the Modeling and Evaluation of Reliability and Availability Growth", *IEEE Trans. on Software Engineering*, vol. SE-17, no. 4, pp. 370-382.
- (20) LITTLEWOOD, B. (1981). "Stochastic Reliability Growth: A Model for Fault Removal in Computer Programs and Hardware Design", *IEEE Trans. on Reliability*, vol.R-30, no.4, pp. 313-320.
- (21) LITTLEWOOD, B. and VERRALL, J.L. (1973). "A Bayesian Reliability Growth Model for Computer Software", *J. Royal Statist. Soc. C (Applied Statistics)*, vol.22, pp. 332- 346.
- (22) MELLOR, P. (1986). "Software Reliability Data Collection: Problems and Standards", *Software Reliability: A state of the Art Report*, Pergamon Infotech Ltd., London, pp. 165-181.
- (23) MILLER, L.H. (1956). "Table of Percentage Points of Kolmogorov Statistics", *American Statistical Association Journal*, pp. 111-121, March.
- (24) MUSA, J.D. and OKUMOTO, K. (1984). "A Logarithmic Poisson Execution Time Model for Software Reliability Measurement", *Proc. Seventh International Conference on Software Eng.*, IEEE Comp. Soc. New York, pp. 230-238.

Acknowledgments

This research was supported by the ESPRIT Basic Research Action PDCS (SB, KK, J-CL, BL, SM) while the data collection and extraction was supported by the Alvey SRM project SE-073 (PM, AT).

Towards the MEDLAR Framework*

Jim CUNNINGHAM[†]
Dov GABBAY[‡]
Hans Jürgen OHLBACH[§]

Abstract

This is an outline description of work seeking an integrated framework for mechanising nonclassical logics. The particular logics and calculi we are concerned with are structured from the point of view of applications. As a first example for testing our prototype of the general framework, a generalised interpretation of modal logics is presented. Next, we introduce the methodology of *Labelled Deductive Systems*, demonstrating why this approach for a general framework is adequate to integrate various logical systems via a unified methodology. Finally, the need for different operational methods of solving problems in formal logic are briefly discussed in the context of an ambitious example suggested from MEDLAR case studies.

1 The MEDLAR Vision

Within the context of MEDLAR, we have two main objectives, namely:

1. “characterise a class of logics capable of handling central problematic aspects of practical reasoning with time, action, belief and intent;”
2. “develop good techniques for mechanised deduction with the logics of practical reasoning.”

The aims of the first year of the project were to choose and analyze some diverse case studies, and to survey current available techniques of representation and deduction.

The emerging new logics (the logics of practical reasoning) are specially designed to address phenomena of particular application, such as belief, modality, time, action planning, intent, and norm. Currently, logical systems capable of representing human interactions are put forward in an ad hoc manner using techniques from classical logic, its proof theory and automated deduction systems. The consortium felt it time to systematise and expand this growing area of practical reasoning and put it in a general framework with firm foundations.

The following unique points characterise our position:

*Research supported by the MEDLAR ESPRIT (3125) Basic Research Action (Mechanising Deduction in the Logics of Practical Reasoning), CEC.

[†]Department of Computing, Imperial College, London SW7 2BZ, UK.

[‡]Department of Computing, Imperial College, London SW7 2BZ, UK.

[§]Max Planck Institute for Computer Science, D-6600 Saarbrücken, Germany.

1. Automated deduction methods are logical as well as computational and form an integral part of a logic. The practical reasoning agent is more sensitive to the deductive method employed than to the consequence relation.
2. The relationship between two sets of formulas Δ_1 and Δ_2 as perceived in the mind of the practical reasoner is not just that of provability ($\Delta_1 \vdash \Delta_2$), but one of many diverse, equally important possibilities. For example, Δ_1 may be a set of axioms and Δ_2 may be a set of facts about the particular situation to be explored. Or, Δ_1 may be a set of hypotheses to be found in order for the set of facts Δ_2 to be true in a theory.
3. In a system of practical reasoning the activity of reasoning is not only performed on the basis of assumptions and conclusions, but it also involves features such as how the conclusions are obtained, how and by whom the queries were asked, etc. There is 'reasoning about the reasoning', and the resulting system is a mixture of the 'levels', so to speak. It is therefore clear that the handling of meta-level features are as important a component of a logic of practical reasoning as its proof theory and theorems; thus, we need to distinguish and control the use of inferential processes so that we can choose (a) (conventional) proof, (b) symbolic execution, (c) abduction, etc.

Recognition that the rôle of automated deduction in systems of practical reasoning is not only computational but also logical has far reaching conceptual and practical consequences [15]. It means we can build our systems on top of existing theorem proving machinery. It means that logics and proof-theoretic techniques are more organically intertwined.

The traditional view is that a logical system is just a consequence relation ' \vdash '. Thus a logic L is essentially the set theoretical relation ' $A_1, \dots, A_n \vdash_L B$ ' (or ' $A_1, \dots, A_n \vdash_L B_1, \dots, B_m$ ') between formulas of the logic. A proof-theoretical system for L is just a way of presenting the consequence relation. A mechanised automated deductive system for L is a computational presentation of L (which is always sound but is rarely complete). Thus we have three layers of systems:

1. The set theoretical logic L ; e.g. classical logic as a set of tautologies;
2. A proof theory for the logic. For example Gentzen systems, resolution systems;
3. An automated implemented deduction refinement for L . For example the connection graph method for doing resolution; a Prolog implementation of tableaux; etc.

The traditional view does not see (2) and (3) as necessarily logical steps. A Resolution System for classical logic and a Gentzen System for classical logic are not perceived as different logics but as two syntactical proof theories for the 'same' logic.

In order to deal with logics of practical reasoning this view has to be modified. A logical system must also contain its proof theory. Thus, classical logic via resolution is a different logic than classical logic via natural deduction. The reason we adopt this approach is because there is evidence in the logics of practical reasoning that method of reasoning (i.e. proof theory) is more dominant in the mind of the reasoner than consequence relation. These views have surfaced in the work of several members of the consortium. Gabbay

in his Goal-directed theorem proving for various intermediate logics has stressed that slight variations on the technique can give diverse logical systems and that humans are much more receptive to reasoning with implications than with disjunctive normal forms. Ohlbach has used two sorted classical logic and modified resolution on the additional sort to obtain a variety of modal logics [21]. His technique is important also because he can modify existing machines to do the job. L. Fariñas del Cerro has developed syntactical resolution systems for modal and temporal syntax which all have the same form except that the notion of a resolvent is generalised and changes from logic to logic. We have here several general proof techniques which can take slight variations to yield different logics. This we consider very important in the area of practical reasoning. The human reasoner is not theorem or tautology conscious. He/she understands modes of reasoning. He/she is more perceptive to how one is arguing than to what the logical theorems are.

The story so far can be summarised by saying that for the logics of practical reasoning the view that a proof technique is part of the logic (and not just the logic as a set of theorems) is natural and essential. However, it still leaves the view that logics deal with deduction and proof, and, given a database Δ and a query Q , that the basic question we are interested in is 'does $\Delta \vdash Q$ in some logic L ?'

An important observation came from one of our members. R. Demolombe says that in the logics of practical reasoning the above question is only one of many questions, which the human practical reasoner may ask [10]. For example, he/she may ask: what do we need to add to Δ in order to prove Q , or, in how many ways can we get Q . Traces of this sort of problem do exist in the literature. There are Prolog programs telling us why queries fail or succeed. There are also programs in planning systems which give us a sequence of actions leading to success. Different ideas of deduction are also used in the linguistic theory of relevance and in engineering design and fault checks. The consortium's view is that we have to make these tasks systematic and incorporate them into a unified logical framework.

In searching for a framework suitable for practical reasoning the above components affect our perception of the relative importance of various logical techniques. The emphasis on methods of translations, interpretation and meta-level reasoning is extremely important. In fact, a meta-level discipline of how one logic can describe in itself the operations of another logic, is a third component in our logical theory of practical reasoning. The representation and propagation of meta-level features within a single logic (i.e. how a single logic can reason about part of itself as a step in its own reasoning) is an essential part of a logic of practical reasoning and an essential perceptible (cognitive) feature of the practical reasoner. Thus, we have the formulas of the logic, the calculus of deduction, and the operational control features which determine how we use the deduction calculus for the problem at hand – whether a theorem prover for mathematics [25], a user modelling system [20], a plan recognition system [7], a deductive planning system, an expert system, etc. –.

The last feature of paramount importance is semantics. Here we agree with the traditional view, but note that it has to be modified to accommodate the new proof techniques and meta-level features. More precisely, we must give semantics not to a logic as a set of theorems (and prove a completeness theorem), but we must give semantics to 'proof techniques,' by explaining the semantics of such notions as 'disciplines for handling labels,' 'disciplines for propagating meta-level features,' etc.

Where does the old notion of a logic fit into this view? We replaced the notion of a logic as a primary notion by the notion of proof technique. Variations on a proof technique yield different logics. Thus, a single logic can be identified by its place in different proof techniques. In practical terms this means that in order to present the old notion of a logic we need to present a system of several logics and their proof techniques and variations, and indicate how other logics reside in the network. This corresponds to identifying a point in space as the intersection of three different planes. We can no longer present a logic in isolation but as the intersection of several proof systems, meta-level features and other human oriented ‘parameters’ involved in practical reasoning. This view should be compared with the more traditional one that a logical system is a consequence relation (a set of theorems) with possibly a proof theory (e.g. Hilbert style axiomatisation) and semantics.

To summarise our position, we claim that for the presentation of the logics of practical reasoning we need (at least) the following components:

1. Logic presentation discipline
 - a. Formalisation techniques: how to formalise an intuitively understood practical reasoning system;
 - b. Proof methods: either on the formal representation of the logic or directly on the structure of application area;
2. Metalevel feature handler: a module which can act as a meta-level system to virtually any logic once the formalisation of the logic is ‘plugged in’;
3. Database query disciplines (provability, abduction, explanation, negotiation, consistency, etc.);
4. General Semantics, which can be broadly specialised for arbitrary logics;
5. Implementation of logical deduction with possibilities of networking different practical reasoning systems.

Before continuing with our theory, we would like to show a small example which illustrates the kinds of practical reasoning we have in mind. After this, we shall look at the structure of domain-oriented logics used in practical reasoning, giving an example of a particular class of logics for which we have developed an appropriate framework. The example is concerned with a general modal logic which deals with a wide range of concepts from epistemic notions to probabilistic reasoning. Finally, the methodology of *Labelled Deductive Systems (LDS)* is introduced. Since *LDS* integrate various aspects of logics and deduction calculi, they emerge as our favourite candidate for a framework to handle the logics of practical reasoning. We conclude by illustrating with an industrial example that all the mechanisms aforementioned are needed in order to mechanise more complex automation tasks.

2 An Example of Practical Reasoning

The small example in this section is intended to demonstrate some of the phenomena which occur in practical reasoning and some kinds of reasoning mechanisms which are

the customer $W_C buy(C, A)$ are almost identical. The only difference is between BMW and A , i.e. the customer does not yet want to buy the BMW . In order to reach his goal, the car seller must achieve $A = BMW$ in the database of the customer. That means we need a mechanism which computes a kind of transformer that maps the second W_C -expression to the first one. Schematically the rule is:

$$\begin{array}{ll} W_a W_b Q & (a \text{ wants } b \text{ to want } Q.) \\ B_a W_b Q' & (a \text{ believes that } b \text{ wants } Q'.) \\ \text{Compute a transformer } \alpha: Q' \mapsto Q & \end{array}$$

It is not yet clear what this transformer is logically. In this example it is simply $A \mapsto BMW$. Using this transformer we can start a process that resolves conflicts which hinder the customer to apply the transformer (“car seller: You could take the BMW ”).

After the customer has uttered the first sentence “I do not want a catalyst” we must insert:

Assumptions about the customer: 8. $W_C \neg cat(A)$

The conflict resolution process now finds a contradiction between 8 and 1 if the transformer $A \mapsto BMW$ is applied. Therefore it generates the next subgoal: Find something in the private beliefs of the car seller which, when told to the customer, implies $\neg W_K \neg cat(A)$. That means, prove the goal:

private beliefs \wedge general knowledge \wedge assumptions about the customer $\Rightarrow \neg W_C \neg cat(A)$.

and tell the axioms in ‘private beliefs’ which are used in the proof to the customer. Since with the current state of the database this attempt fails, the conflict resolution process tries to find possible reasons for the ‘wants’ which cause the conflict. That means: it tries to figure out why $W_C \neg cat(A)$ holds in order to remove these causes. In this case, no reasons can be found therefore the car seller must ask: Why do you not want a catalyst?

The answer, “because it reduces the engine power,” in a logical representation is:

$$W_C \neg cat(A) \text{ because } B_C(\forall c : Car \ cat(c) \Rightarrow rep(c))$$

This formula has to be inserted into the assumptions about the customer. Now we need an inference rule which allows deducing further reasons. The rule we need says intuitively: If you do not want R because you believe that R implies Q then in reality you do not want Q .

$$\frac{W_x \neg R \text{ because } B_x(R' \Rightarrow Q) \quad \sigma R = \sigma R'}{W_x \sigma \neg R \text{ because } W_x \sigma \neg Q}$$

and in the concrete example:

$$\frac{W_C \neg cat(A) \text{ because } B_C(\forall c : car \ cat(c) \Rightarrow rep(c))}{W_C \neg cat(A) \text{ because } W_C \neg rep(A)}$$

The conclusion, together with the assumption $W_C \neg rep(A)$, is inserted in the set of assumptions about the customer.

The reason $W_C \neg rep(A)$ for not wanting a catalyst can not yet be removed. Using the transitivity of the *because* relation we can infer from this and axiom 3 a deeper reason:

$$W_C \neg cat(A) \text{ because } W_C \text{ Can drive-fast}(C, A)$$

W_C *Can drive-fast*(C, A) is now a reason for not wanting a catalyst we can remove using axioms 2,4 and 6 which allows to derive $\neg W_C$ *Can drive-fast*(C, A). From the proof we take that 6: $F \forall x : \text{car speed-limit}(x)$ is a private belief of the car seller. Therefore this is the missing information the seller must give to the customer such that the customer can infer that he needs no fast car, and therefore there is no further reason for not wanting a catalyst.

Let us briefly review what we have discovered in this exercise. For a start we have addressed an area of practical reasoning which at present lies wholly within the realm of human competence. In addressing the area we have discovered a logical form with many modalities and inference rules: the essence of a logical framework. What we do not have is a clear picture of the way to control its operation. We need these extra meta-level elements.

3 Structuring the Area

When considering the application of formal logic to solve a particular problem of practical reasoning one is likely to first identify a basic underlying logic (classical, intuitionistic, relevant, etc.), find its specific manifestation (perhaps with one or more modal features), determine the operational model, and seek in turn to implement it. Thus, we may loosely identify the steps as:

- Step 1: Identify the basic logic(s).
- Step 2: Identify the specific logic(s).
- Step 3: Identify the operational model.
- Step 4: Implement the application.

Decisive questions are likely to arise in the first couple of steps, such as, for example: Is predicate logic sufficient? Does sentence truth depend on contexts, worlds, or states? What kind of implication is needed: material, relevant, intuitionistic, etc.? Are there nonmonotonic effects, defaults, probability values? Are there problems requiring special treatment, for example taxonomic reasoning in the KL-ONE style [2]? How much is logical reasoning and computation in special models, for example in geometric reasoning, interrelated? Each of these aspects has almost the same impact on the final realisation of the application as questions like ‘Is a database necessary?’ or ‘Is sequential computing adequate?’ or ‘What is the right programming paradigm: conventional, object oriented, rule based etc.?’ have on the realisation of conventional software projects.

Looking at the landscape of logics as it stands, it looks like a patchwork of mostly very loosely correlated concepts. Classical predicate logic is the formalisation of the logical connectives as truth functions, together with total functions and relations appropriate to the application. Modal logics extend predicate logic in some sense, though (in most versions) not fully, by introducing the concepts of states (worlds) and state transitions (accessibility relation). There are logics like linear logic, relevance logic and intuitionistic logic, which have a weaker notion of implication. There are probabilistic logics for dealing with uncertainty. There are nonmonotonic and default logics. There are very special systems like KL-ONE for representing taxonomic hierarchies, or polynomial algebras which can be used very successfully for geometric reasoning.

We see two main aspects for distinguishing all these systems. First of all there may

be concepts which cannot be described conventionally using classical predicate logic. Examples of these are the concept of states and state transitions, of nonmonotonicity and default values, and different notions of implication. The second aspect is that some fragments of logic have been identified which have interesting applications and yet allow for a more efficient algorithmic treatment. Examples are equational reasoning with completion methods, or polynomial algebras with Gröbner base method [3], or PROLOG Horn clauses, or unary and binary predicates in KL-ONE algorithms.

In developing an integrated framework, the efficiency argument of the second aspect is a good reason to keep these fragments still identifiable such that they can be handled in the best possible way. That means we need an extension of predicate logic where the special systems can keep their identity. A prototype of such a system which combines general predicate logic with a very special treatment of a certain fragment of logic is order sorted predicate logic. The idea is to represent unary predicates and some correlations between them not as formulas, but in a sort structure where special algorithms operate. For example, the recursive formula $\forall x P(x) \Rightarrow P(f(x))$ that allows us to derive from $P(a)$ infinitely many $P(f(\dots f(a)\dots))$'s is represented as a sort declaration $f : P \mapsto P$. From this, nothing can be derived, but nonetheless if a has sort P then the sort P can be computed for arbitrary nestings of $f(\dots f(a)\dots)$. Thus the recursivity has been shifted from the object level into the meta level of the sort computation. A resolution calculus system based on this combined logic in general behaves much better than in the unsorted case. In [24] this idea has been put forward to more general cases.

Recently constraint systems [4] have emerged as a conceptual framework for combining predicate logic with calculi and algorithms of specialised theories. On the more general level, combination of, say modal logic, relevance logic etc., we shall see that *Labelled Deductive Systems* provide the appropriate mechanisms.

Having decided about the basic logic (or the combination of logical disciplines such as, e.g. modal-relevance-default logic), the particular logic(s) has to be identified in the second step. If, for example, first order predicate logic is sufficient, the particular application may need only Horn clauses, which allows for a much simpler and efficient algorithmic treatment. If, however, the possible worlds approach has been selected, the next step is to choose among standard relational Kripke semantics or minimal model semantics and further refine to choose among the particular logics, say K, T, S4, S5 etc [6]. Since in many applications, these special logics occur simultaneously, for example S4 for expressing 'belief' and, say, K for formalising the 'want' operator, a general framework on this level is also badly needed. In the next chapter we present the ideas for combining these systems.

Having understood a logic and developed a deduction calculus (e.g. resolution, tableaux, etc.) is only the first step towards mechanising it on a computer. In the third step, the identification of the operational model, we have to analyze what really needs to be computed in our application. Possible operational models are:

1. theorem proving: check whether a theorem follows from a set of axioms.
2. abduction: find additional assumptions sufficient to prove a theorem from given assumptions.
3. model generation: find a model that satisfies a given statement.

4. synthesis: find constructive proofs for existentially quantified statements. (Logic programming and deductive planning, for example, fall into this category.
5. closure computation: generate all consequences from a given set of formulas.

For each of these reasoning methods there are many different ways to implement them on computers. In contrast to the general logical framework we are working on, we do not believe on *the* ultimate inference method that solves all problems in the given class with the same efficiency. (If there existed one, one would teach it in the elementary school and get perfect mathematicians.) Most of the work done in the area of automated deduction in the past has been to develop methods for special subclasses of problems. Within the MEDLAR project we have also done work on special areas of reasoning. For example, model construction [5], and geometric reasoning (in [10]).

Although there cannot be the ultimate reasoning method, what we should aim for is an environment where all the different methods can cooperate. It might be too early to report here about our ideas and results on combining different reasoning methods, but it would perhaps be worth mentioning that some interesting applications of our prototype of general framework are proving very interesting indeed (e.g. the recent account of linguistic relevance, abduction and analogy via *Labelled Deductive Systems* reported in [17, 16]).

In most applications of logic so far only very small fragments of the huge variety of possibilities logic offers, are exploited. Usually, one particular logic and one particular inference method is chosen. With more complex applications more logics and operational methods are needed simultaneously. For achieving the ultimate goal of Artificial Intelligence, realising autonomous intelligent agents which are able to interact flexibly with their real world environment, everything logic can offer is badly needed. That means isolated solutions – one particular logic with one particular calculus that can be implemented to perform one particular job – is not what we need. We need integrated solutions and structuring methods where many different logics can be combined in one system, and that the knowledge represented in that system can be manipulated in various ways. This is an area of active work within the project.

4 Modal Logic

We now sketch our work on modal logics as an example for a framework for a particular class of logics which is relevant in the second step mentioned in the previous chapter: identify the specific logic(s).

Modal logics with possible worlds semantics have been used for formalising situations where states and state transitions are basic phenomena, as temporal logic where the flow of time changes the states, action logic where actions cause transitions between states, epistemic logic and doxastic logic where transitions to imaginary worlds in some agent's mind occur, deontic logic with transitions to 'ideal' worlds etc. Since the interpretation of the \Box -operator as *necessarily, knows, believes, ought, henceforth* is completely irrelevant from a logical point of view, this is already a considerable unification of many different concepts. For particular interpretations of the \Box operator, however, we must impose particular properties on it.

For example, if you want the \Box -operator to be interpreted as 'knows' and you want to build into the logic that nobody can know false things, you add the formula schema $\Box \mathcal{F} \Rightarrow$

\mathcal{F} , i.e. if somebody knows \mathcal{F} then \mathcal{F} is really true. If you want to model introspection, i.e. the agents know what they know, you add $\Box\mathcal{F} \Rightarrow \Box\Box\mathcal{F}$. If you want to infer from a tautology $\mathcal{F} \Rightarrow \mathcal{G}$ the formula $\Box\mathcal{F} \Rightarrow \Box\mathcal{G}$ you add this as an inference rule in the Hilbert calculus. For a realistic ‘knows’ operator this might not be appropriate because it means a kind of logical omniscience. If for example $\mathcal{F} \Rightarrow \mathcal{G}$ is ‘from Zermelo Fränkel axioms for set theory, Fermat’s last theorem follows’ (assume this is so) then knowing the axioms for set theory automatically implies knowing that Fermat’s last theorem is true. This is obviously too strong. On the other hand, if \Box is meant to be an action operator, i.e. $\Box\mathcal{F}$ means ‘after performing an action, \mathcal{F} holds, this rule is perfectly okay.

From an application point of view it is therefore desirable to have a system where such formula schemas and inference rules can be turned on and off just by a simple declaration. Furthermore, it is necessary to apply these schemas selectively. For example, there might be two agents, say Tom and Mike, Tom being an expert who does not know false things, i.e. the schema $\Box\mathcal{F} \Rightarrow \mathcal{F}$ holds for Tom. For Mike, however, this may not be sure so we do not want this schema for him. To distinguish them, the modal operator has to be parameterised with the agents. That means instead of $\Box P$, $[Tom]P$ or $[Mike]P$ respectively is written. Now the schema $[Tom]\mathcal{F} \Rightarrow \mathcal{F}$ declares that Tom is an expert, but nothing is known for Mike. Having not only constant symbols as parameters of the modal operators, but arbitrary terms, we can distinguish the different interpretations of the operators when they occur simultaneously. For example, $[want\ Tom][know\ Tom]time$ can be used to express: Tom *wants to know* what time it is². This may be a logical representation of Tom’s question ‘what time is it?’. Since intentions (want-operator) and knowledge (knows operator) have really different characteristics it is again obvious that the axiom schemas and inference rules have to be specified selectively. One idea, to take a second order logic, where a formula schema is just one formula among others, presents us with the problem that there is no established calculus for second order modal logic. This problem has been addressed by workers within MEDLAR [19] seeking adequate extensions of the Curry-Howard functional interpretation for a wide range of modal logics.

An alternative, which immediately yields an efficient calculus has been presented in [22]. The idea consists of two parts. First of all the possible worlds semantics of modal logic is exploited to get rid of the operators by translating them into standard predicate logic. Then the well known correspondences of certain modal axiom schemas with properties of the possible worlds structure are used to replace the schemas with axioms talking about the corresponding semantic structure. Consider for example the formula $\Box\Diamond P$ whose interpretation is ‘for all worlds, say w_1 , accessible from the initial world (\Box -operator) there is a world, say w_2 , accessible from w_1 (\Diamond -operator) such that P holds in w_2 . This formula is translated into $\forall x \exists y P'(x \circ y)$. x and y denote *transition functions*, i.e. functions which map worlds to accessible worlds. For a particular x and a particular y , $x \circ y$ therefore denotes a function that can be applied to the initial world and may return for example the world w_2 . Then $P'(x \circ y)$ requires looking whether P holds in w_2 . A slight extension handles the case that the operators are parameterised. For example, $[want\ Tom][know\ Tom]time$ is translated into $\forall x \forall y time'(x(want(Tom)) \circ y(know(Tom)))$. Now these functions which map worlds to accessible worlds are themselves parameterised with the parameters of the modal operators such that, for example, the interpretation of $x(want(Tom))$ yields

² W_{Tom} or $\Box_{W(Tom)}$ are syntactic alternatives.

a normal function for mapping worlds to accessible worlds.

This 'functional' translation method allows us to get rid of the operators and to generate very compact predicate logic formulas. The next step is to translate the characteristic axiom schemas. For many axiom schemas there is a correspondence with certain properties of the underlying semantic structure. For example, $\Box\mathcal{F} \Rightarrow \mathcal{F}$ holds if and only if the accessibility relation is reflexive. Reflexivity of the accessibility relation, however, can be axiomatised with an equation over transition functions. In this case, the axiom needed expresses just that the identity functions which maps worlds to itself is a transition function, i.e. $x \circ id = x$. That means the *axiom schema* $\Box\mathcal{F} \Rightarrow \mathcal{F}$ has been replaced with an *axiom* and this equation is in fact so simple that it can be turned into a theory unification algorithm [21]. And in fact, turning an axiom schema into a theory unification yields the optimal treatment.

Those axiom schemas we can handle, that means where we know the semantic counterpart and can axiomatise it in first order logic, can be embedded through translation into more complex formulas yielding new versions we can also handle. For example

$$Q \Rightarrow (\forall\mathcal{F} \Box\mathcal{F} \Rightarrow \mathcal{F})$$

which makes the reflexivity of the accessibility relation depending on some condition, is translated into the axiom:

$$Q \Rightarrow (\forall x \ x \circ id = x).$$

That means we have a much more sophisticated tool for adjusting the meaning of the operators to the given application.

We increase the number of basic axiom schemas we can handle by identifying the corresponding semantic property and axiomatising it in the translated logic. If the axioms are simple enough they can even be turned into theory unification algorithms. Previous reports [21, 22, 23] give an overview about the axiom schemas we are able handle so far. By extending this method to the logic with parameterised operators we obtain an optimal treatment of a very complex logic where almost everything that can be done with possible worlds logic is brought together.

From a practical point of view, however, a lot of work has still to be done. The axiom schemas which occur in recent applications of modal logic are a magnitude of order more complex than those we have investigated so far. To give an example, in a plan recognition scenario we might want to formalise the following rule.

If agent *A* believes that action α very likely brings about \mathcal{F} and *A* actually wants to achieve \mathcal{F} then there is certain evidence that *A* intends to do α .

Expressed in modal logic with parameterised operators this rule reads roughly

$$\begin{aligned} & ([believes\ A][action\ \alpha][probability\ very-likely]\mathcal{F} \wedge [want\ A]\mathcal{F}) \\ & \Rightarrow [probability\ some-evidence][want\ A]Do(\alpha) \end{aligned}$$

or schematically

$$([p][q][r]\mathcal{F} \wedge [s]\mathcal{F}) \Rightarrow [t][s]Do(\alpha)$$

This is again an axiom schema, but so far we do not know how to translate it into an axiom. Quite a number of schemas of this kind occur for example in [8]. We are confident that there is a way to cope with most of them in the described way.

The translation method allows us all the modal axiom schemas which can be translated into predicate logic axioms to occur as subformulas in modal logic axiomatisations. Therefore we get a really expressive logic where many modalities with different and really complex characteristics may occur simultaneously. A statement which can easily be expressed in this logic is for example: “naive people believe that computers are infallible”:

$$\forall x : \text{people naive}(x) \Rightarrow [\text{believe } x] \forall y : \text{computer } \forall \mathcal{F} [\text{believe } y] \mathcal{F} \Rightarrow \mathcal{F}$$

Using the translation technique which replaces the formula schema with a formula, standard predicate logic inferences can for example deduce from the facts that Tom is naive ($\text{naive}(\text{Tom})$) and that he believes that a certain computer believes that he has to pay tax ($[\text{believe } \text{Tom}] [\text{believe } \text{computer}] \text{has-to-pay-tax}(\text{Tom})$) that he really has to pay tax ($[\text{believe } \text{Tom}] \text{has-to-pay-tax}(\text{Tom})$.)

One step further has been achieved recently by unifying modal logic with probabilistic logic [23]. The semantics we use for this logic is a modification of neighbourhood semantics (minimal model semantics) where instead of an actual world we have an *actual world set* and the neighbourhood structure is computed by an *access function* $\varphi(p, W)$ which computes for a parameter p of the modal operator (or its interpretation respectively) and the actual world set a set of sets of worlds. If for example $[\text{.5}] \text{raining}$ is to be interpreted as “in at least (at most, exactly) 50% of all cases (worlds)” it is raining, then $\varphi(\text{.5}, W)$ computes all at least (at most, exactly) 50% subsets of the actual world set W . $[\text{.5}] \text{raining}$ hold in the actual world set if *raining* is true in all worlds of one of these subsets.

With this version of neighbourhood semantics we get a very weak and therefore very flexible logic, where the only built in law is: If $\mathcal{F} \Leftrightarrow \mathcal{G}$ is a tautology then so is $[p] \mathcal{F} \Leftrightarrow [p] \mathcal{G}$. Specifying the properties of the access function φ for each parameter now permits the interpretation of the $[p]$ -operator for instance as probability operator (with numbers or symbolic values as probabilities), actions (dynamic logic), agents who believe, know, want etc. things, as temporal operators where for example $[\text{6h}] \text{shining}(\text{sun})$ means that the sun is shining in an interval of 6 hours. Very straightforward are further possibilities for combining concepts. For example, $[\text{believe } \text{Tom } \text{.8}] P$ is an allowed formula with the intended meaning “Tom believes P with 80% confidence”.

We have investigated characteristic operations on the parameters of the modal operators and correlated them with characteristic axiom schemas. For example, the axiom schema $[p][q] \mathcal{F} \Rightarrow [p * q] \mathcal{F}$ introduces the $*$ -operation on the parameters. In the probabilistic interpretation where p and q are numbers, ‘ $*$ ’ is just the multiplication of numbers. $[p * q]$ is the overall probability of nested probabilities. In the action logic interpretation where p and q denote actions, ‘ $*$ ’ means sequencing of actions. Exactly the same axiom schema holds, and therefore ‘ $*$ ’ is independent of the actual application of the logic.

As a further example, the axiom schema $p \leq q \wedge [q] \mathcal{F} \Rightarrow [p] \mathcal{F}$ specifies an ordering relation \leq on parameters. Depending on the application as probability logic, action logic, temporal logic etc., it expresses an ordering on probability values, a specialisation hierarchy of actions, a subinterval relation etc.

A number of other operations have been investigated with respect to their underlying logical properties and their interpretation in various applications of the logic. It turns out that we get a very general theory which includes everything that can be done with possible worlds semantics and possibly much more. Some other logics for which a possible worlds semantics has not been known so far have shown to be specialisations of the framework

[11]. We believe that with this approach we have a true generalisation of the possible worlds framework.

5 Labelled Deductive Systems

The methodology of *Labelled Deductive Systems* [13, 14] gives us a general framework for the formalisation of many aspects of practical reasoning. The approach replaces the traditional notion of consequence relation between formulas of the form $A_1, \dots, A_n \vdash B$ by the notion of consequence relation between labelled formulas as in:

$$t_1 : A_1, t_2 : A_2, \dots, t_n : A_n \vdash s : B$$

As we have said earlier, the rôle of the label is to provide additional information about the formula which is not of the same declarative nature as that of the formula itself. The label t in $t : A$ can represent the degree of reliability of the item of data A , or can be a λ -term representing a proof of A , or as in the case of many valued logics, t can indicate the range of truth values of A . Thus, depending on the logical system involved, the intuitive meaning of the labels vary. In querying databases, we may be interested in labelling the assumptions so that when we get an answer to a query we can indicate, via its label, from which part of the database the answer was obtained. Another area where labelling is used is temporal logic. We can time stamp assumptions as to when they are true and, given those assumptions, query whether a certain conclusion will be true at a certain time. The consequence notion for labelled deduction is essentially the same as that of any logic: given the assumptions, does a conclusion follow?

This approach seems to be successful in unifying and formalising practical reasoning because it has as its basic components and basic logical moves the same features which appear in practical reasoning inference. The practical reasoner deals with declarative information: this is formalised as the formula A . He/she also has other information about A which he/she keeps to himself/herself. For example, 'how reliable A is', 'what context A is in', 'under what conditions is A to be applied', 'who told him/her about A ', etc. When reasoning with formulas like A the extra information gets manipulated along with the formulas. Since *LDS* officially allows for labels alongside formulas, one can mirror in the *LDS* methodology the natural moves of the practical reasoner. In traditional logics without labels, one can either ignore the label and be faced with shortcomings (e.g. that the extra information is indeed necessary), or incorporate the extra information into the formula A to form A^* at the cost of making the logic cumbersome and complex.

The labels allow us to code meta-level information. For example, if we want to reason about 'our proof so far', we can either go to a meta-logic which names proofs and talks about them, or we can tag (label) formulas and propagate the tag, coding the necessary meta-level information in the tag. In Computer Science terms this would be identified as some sort of implementation. Of course, in *LDS* it is done logically.

Whereas in the traditional logical system the consequence relation is defined using proof rules on the formulas, in the *LDS* methodology [13] the consequence relation is defined by using rules on both formulas and their labels. Formal rules are then established for manipulating labels and this allows for more scope in decomposing the various features of the consequence relation. The meta-level features coded by the extra tag aforementioned

can be formally reflected in the algebra or logic of the labels and the object-level logical features can be reflected in the rules operating on the formulas.

At this point the reader may worry that the labels can be used to code anything. This is not the case.

The manipulation of the labels is not arbitrary, rather it follows rigorous disciplines inherent to the logics being dealt with. For example, for the case of implication ' \rightarrow ', the following equation must be satisfied:

$$t \text{ labels } A \rightarrow B \text{ iff } \forall x \in \mathbf{L} [\text{If } x \text{ labels } A \text{ then } t + x \text{ labels } B].$$

\mathbf{L} is a set of labels and '+' is a binary associative operation. Different logics have the freedom of choice of \mathbf{L} and '+' can be numerical addition (for Lukasiewicz' logics), λ -functional application (for intuitionistic logic and the Curry-Howard interpretation [18]) or multiset union (for Girard's linear implication). For non-monotonic reasoning systems, the labels could be formulas of some underlying monotonic system and '+' could represent some logical operations in the underlying system.

To do modal logic in *LDS*, the essential idea here is that the possible worlds become labels. To show how it is done, let the language contain propositional variables p, q, r, \dots , the classical connectives $\wedge, \vee, \rightarrow, \perp$ (a constant denoting *contradiction*) and the modalities \Box and \Diamond which may here be read in the temporal sense of *henceforth, it will always be the case that*, and *sometime in the future it will be the case that*. The labels are atomic labels from a set $\{t_1, s_1, t_2, s_2, \dots\}$ and we have the configuration relation $<$ on labels. A database has the form of a set of labelled formulas called assumptions and a configuration on the labels involved in the assumptions. Consider the database:

Assumptions	Configuration
(1) $t : \Box\Box B$	$t < s$
(2) $s : \Diamond(B \rightarrow C)$	

The conclusion to show (or query) is:

$$t : \Diamond\Diamond C.$$

The derivation is as follows:

3. From (2) create a new point r with $s < r$ and get $r : B \rightarrow C$.

We thus have

Assumptions	Configuration
(1), (2), (3)	$t < s < r$

4. From (1), since $t < s$ get $s : \Box B$.
5. From (4) since $s < r$ get $r : B$.
6. From (5) and (3) we get $r : C$.
7. From (6) since $s < r$ get $s : \Diamond C$.
8. From (7) using $t < s$ we get $t : \Diamond\Diamond C$.

Discussion: The object rules involved are:

$\Box E$ Rule:

$$\frac{t < s; t : \Box A}{s : A}$$

$\Diamond I$ Rule:

$$\frac{t < s, s : B}{t : \Diamond B}$$

$\Diamond E$ Rule:

$$\frac{t : \Diamond A}{\text{create a new point } s \text{ with } t < s \text{ and deduce } s : A}$$

Note that the above rules are not complete. We do not have rules for deriving, for example, $\Box A$. The rules are all for intuitionistic modal logic.

The meta-level consideration may be properties of $<$,

eg $t < s \wedge s < r \rightarrow t < r$ or

eg linearity: $t < s \vee t = s \vee s < t$ etc.

In the example, the deduction rules allowed us to move from one database and configuration to another, obtained by adding assumptions and labels and extending the configuration according to certain rules. The rules used are the rules of the particular *LDS* system for the above language.

It is possible in many cases to translate *LDS* into classical logic. For a formula and a label of the form $t : A$ we write $A^*(t)$, a predicate in classical logic. The translation inductively follows the formation rules of A and the label manipulation rules become axioms of classical logic. E.g.:

1. $(t : A)^* = A^*(t)$ for A atomic.
2. $(t : A \wedge B)^* = A^*(t) \wedge B^*(t)$.
3. $(t : A \rightarrow B)^* = A^*(t) \rightarrow B^*(t)$.
4. $(t : \perp)^* = \perp$.
5. $(t : \Box A)^* = \forall s(t < s \rightarrow A^*(s))$.
6. $(t : \Diamond A)^* = \exists s(t < s \wedge A^*(s))$.
7. $(t < s)^* = t < s$

6 Specific Logics

Summarising what we have discussed so far, we can say that we are close to an integrative framework for syntax and semantics that describes almost all phenomena which have been considered in logic so far. The main ingredients are: a highly general notion of possible worlds, and a general methodology for handling labelled formulas. How do we mechanise this system, i.e. find a calculus that can be efficiently implemented? Here the question arises whether this calculus should work on the original syntax of the logic or whether some kind of translation into an ‘assembler logic’, for example predicate logic is better suited. Experience with compiling programming languages have proved that compilation gains efficiency by rearranging the information (the program) which is organised in a human

oriented style into structures which better suit the underlying processor. A similar effect has been demonstrated for the mechanisation of logics. To illustrate this, consider the following problems:

From the modal formula $\forall x P(x) \Rightarrow \Box P(x)$
it does *not* follow $P(a) \Rightarrow \Box P(a)$

if a is a flexible constant symbol that may change its value from world to world. If for example P is *is-named-Helmut-Kohl*, a is 'chancellor(Germany)' and \Box means 'next-year' it is obvious that from

$$\forall x \text{ is-named-Helmut-Kohl}(x) \Rightarrow \text{next-year is-named-Helmut-Kohl}(x)$$

we cannot derive

$$\text{is-named-Helmut-Kohl}(\text{chancellor}(\text{Germany})) \Rightarrow \text{next-year is-named-Helmut-Kohl}(\text{chancellor}(\text{Germany}))$$

That means *the substitution law does not hold*.

Another problem is that from $a = b \wedge \Box P(a)$ one cannot infer $P(b)$ because the two occurrences of a are in different modal contexts and therefore may denote different things. And that means Leibniz's principle, substituting equals for equals, which is the basis for the paramodulation rule, is not applicable. For most calculi which rely on the substitution rule and Leibniz's principle, this is a disaster.

Melvin Fitting has tried to overcome these problems by incorporating lambda calculus in modal logic [12]. But the problem can be solved much more naturally. The basic problem is that the nesting of modal operators introduces implicitly contextual information. This information is needed for interpreting the symbols occurring in their scope. Making this implicit information explicit by adding additional arguments to the terms and literals permits the distinction of objects which are syntactically identical, but semantically different. We have developed a translation technique which replaces the modal operators by quantifiers over *transition functions*, i.e. functions mapping worlds to accessible worlds. The quantified 'world variables' label the terms and atoms, thus making the implicit information explicit. For example, in the standard relational possible worlds framework, $\Box \Diamond P$ is translated into $\forall x \exists y P[xy]$ where $[xy]$ denotes the composition of two functions which map the initial world to the actual world. In the neighbourhood semantics framework the translation is technically more complex, but the idea is the same.

The counter example for the substitution law we gave above is no longer a problem. The formula $\forall x P(x) \Rightarrow \Box P(x)$ is translated into $\forall x P(x) \Rightarrow \forall y P'(y, x)$ If a flexible term is now to be substituted for x , it carries its modal context, say $a(\text{id})$ (id stands for the identity function). The instantiated formula is $P(a(\text{id})) \Rightarrow \forall y P'(y, a(\text{id}))$. The modal context of the predicate P' , namely y , may now differ from the modal contexts of its arguments, and therefore the two occurrences of a are still the same. This simple translation idea turns out to be sufficient to translate a large class of logics into predicate logic. That means we no longer need specialised calculi and inference systems for each of these logics. Previous work on automated reasoning systems for predicate logic can be applied immediately to them.

A concrete logic can be obtained by specifying the properties of the logical connectives. That means either adding formulas schemas to the set of tautologies and inference rules

of its Hilbert calculus or imposing special conditions to the semantics of the logic. With these buttons one can tune the logic to suit a particular application. For example if you want the \Box -operator to be interpreted as 'knows' and you want to build into the logic that nobody can know false things, you add the formula schema $\Box\mathcal{F} \Rightarrow \mathcal{F}$, i.e. if somebody knows \mathcal{F} then \mathcal{F} is really true. If you want to model introspection, i.e. the agents know what they know, you add $\Box\mathcal{F} \Rightarrow \Box\Box\mathcal{F}$. If you want to infer from a tautology $\mathcal{F} \Rightarrow \mathcal{G}$ the formula $\Box\mathcal{F} \Rightarrow \Box\mathcal{G}$ you add this as an inference rule in the Hilbert calculus.

Obviously, in order to be able to apply deduction techniques to databases involving the necessity modality, it is of paramount importance that one has a system where 'contexts' (structured collections of formulas) can be handled as objects in a similar way that individuals are handled by first-order predicate logic. In the present state of affairs, there is no well-established calculus of second-order logic, and this might seem to be a drawback. Nonetheless, one finds the recent literature many attempts to formulate calculus for second-order logics. Within MEDLAR we are looking at how the methodology of *LDS* and the techniques used in the functional interpretation of logics [18] can be extended to handle structured contexts of formulas and individuals. It is expected that the work will eventually give rise to a parameterised calculus of second-order logics [19].

The translation method we sketched in the previous chapter has offered a new possibility.

7 An Industrial Challenge for the Future

To give a more graphic picture of potential applications, consider the way in which the various logics and styles of reasoning are needed in inventing a factory of the future. Figure 1 shows a robotic car painting workplace with various spacial constraints. It has been suggested by the MEDLAR group at Linz. Initially there are a large number of cars in one workspace (E1) each to be painted a selected colour by one of the robots. A robot is only permitted to paint a car in a place which is free of neighbouring objects, so cars must be moved accordingly. E4 is another space for storing cars, but eventually all cars must be painted and returned to E1.

As sketched, there are a myriad of unresolved questions which we will not go into save to indicate the design space of solutions. It is clear that special geometric reasoning is needed, but the major open questions concern the capability of the robots. This in turn determines the extent to which knowledge and planning is distributed, and the structure of the communication system. Even without autonomous robots there is a considerable problem in finding some partial ordering of tasks which will eventually achieve the goal situation. But this planning problem is precisely the constructive synthesis task mentioned in chapter three. Having autonomous agents makes the conception of a plan more difficult, but does not necessarily complicate the computation of a solution, because with many agents eager to work the search space for tasks of each individual agent can be much smaller. Indeed there are indications that more agents can make the synthesis task more, not less tractable.

Now consider what would happen if we had specified the logical behaviour of each robot, with or without a global planing agent. Whatever our specification, we need to gain confidence in it. One way is to prove some theorems about it, this is traditional deduction. We would like to prove that the goal is eventually achieved, that robots do not

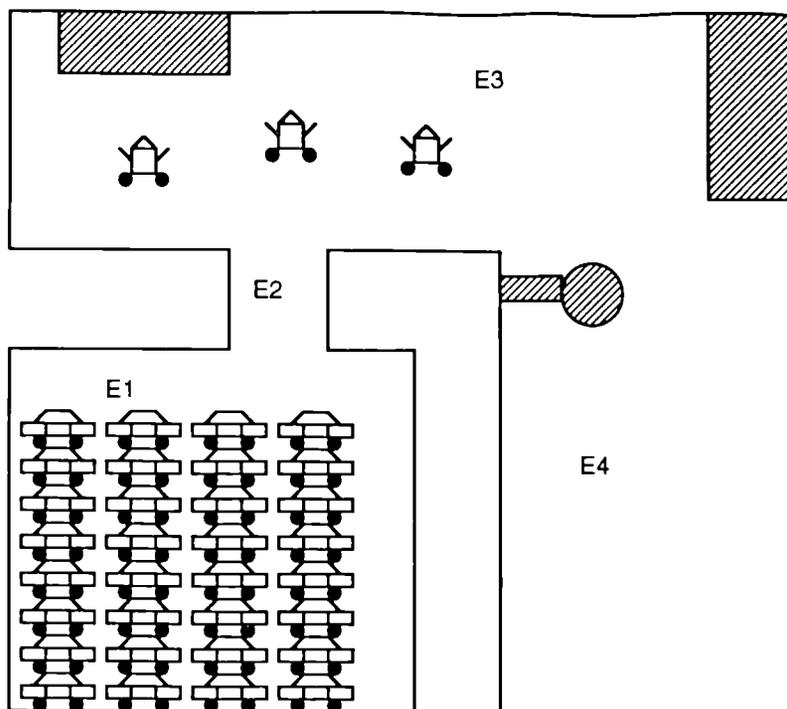


Figure 1: A Robot Car Painting Factory

crash, respray cars or anything else we would not like. (See, e.g. [1] for an indication that this sort of proof is not fantasy.) Obviously, we cannot think of everything, and nor will robots. We might gain more confidence and validate the whole system by simulating it, but if we simulate behaviour faithfully this is a form of model construction, also called symbolic execution and logical animation. This form of reasoning also amounts to interpreting the specification as a very high-level multi-modal logic program. It is a restricted form of closure computation, where we circumscribe the class of consequences which we admit [9].

Finally, we can consider where abduction arises. This is the sophisticated, almost introspective form of reasoning which tells us the additional assumptions needed for a desired conclusion. It has been suggested as a way of giving helpful answers in database interrogation. It is also the form of reasoning which tells us the additional constraints needed when we have some elements of a design, but would like to ensure the required behaviour. This is the type of sophisticated design tool needed for formal design. When an intelligent autonomous car-painting robot perceives another returning a car, it should reason that the car has been painted and so reduce the search space of possibilities for new work.

In conclusion, we have started with a vision, where the large range of formalised systems of logical reasoning about practical situations become available as mechanised tools for human application. To achieve this we recognise that form in deduction is just as important as the traditional content of a logic, its class of truths. By reference to various applications we have sketched some of the analyses which enables us to mix logics and combine systems of reasoning.

Just because we are a basic research action seeking to mechanise practical reasoning it behoves us to re-address some real practical problems in designing the future. Ultimately, we must be judged by applicability.

The logics of real engineering are immensely complicated, but this is a timely challenge because at the present time we already have powerful computing engines available. It is not sufficient for us to merely describe behaviour in a way which admits logical systems of analysis, but we must provide effective methods of reasoning about the function, performance and safety of real engineering systems. That is why mechanising deduction in the logics of practical reasoning is so important.

Acknowledgements. The research reported here emerges from collaboration with all members of the MEDLAR Action. See [10] for work reported at the first milestone. The inspiration for this paper comes from the concluding presentation of Hans Jürgen Ohlbach at the recent Turin Workshop. Ruy de Queiroz has been instrumental in ensuring that the paper exists.

References

- [1] Will Atkinson and Jim Cunningham. Proving properties of a safety-critical system. *Software Engineering Journal*, 6:41–50, 1991.
- [2] R. Brachman and J. Schmolze. An Overview of the KL-ONE Knowledge Representation System. *Cognitive Science*, 9(2):171–216, 1985.
- [3] Bruno Buchberger, G.E. Collins and Berhard Kutzler. Algebraic Methods for Geometric Reasoning. *Annual Review of Computer Science*, 3:85–119, 1988.
- [4] Hans Jürgen Bürckert. *A Resolution Principle for a Logic with Restricted Quantifiers*. PhD Thesis, FB. Informatik, University of Kaiserslautern, 1990.
- [5] Ricardo Caferra and Nicolas Zabel. Extending resolution for model construction. In J. van Eijck, editor, *Logics in AI*, pages 153–168, Amsterdam, 1991. Springer Verlag. Lecture Notes in Artificial Intelligence.
- [6] B. F. Chellas. *Modal Logic: An Introduction*. Cambridge University Press, Cambridge, 1980.
- [7] Philip R. Cohen, Jerry Morgan and Martha E. Pollack (eds.). *Intentions in Communication*. MIT Press, 1990.
- [8] Philip R. Cohen and Hector J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42:213–261, 1990.
- [9] M.C. Costa, R.J. Cunningham and J. Booth. Logical animation. In *12th International Conferencce on Software Engineering*, pages 144–149, Nice, 1990. IEEE Computer Society Press. March 26–30, 1990.
- [10] Jim Cunningham et al (eds). MEDLAR. Milestones deliverables. Internal Report, 1990.
- [11] Luis Fariñas del Cerro and Andreas Herzig. A modal analysis of possibility theory. *Languages et Systèmes Informatique*, Université Paul Sabatier, Toulouse, 1991.

- [12] Melvin Fitting. Modal logic should say more than it does. In Jean-Louis Lassez and Gordon Plotkin, editors, *Computational Logic, Essays in Honor Alan Robinson*. MIT Press, 1991. forthcoming.
- [13] Dov Gabbay. *LDS - Labelled Deductive Systems*. Preprint, Dept of Computing, Imperial College, London, SW7 2BZ, UK. First Draft September 1989, Current Draft February 1991. 265pp. Published as CIS-Bericht-90-22, Centrum für Informations- und Sprachverarbeitung, Universität München, Germany. Final version to appear as a book with Oxford University Press.
- [14] Dov Gabbay. *Labelled Deductive Systems. A Position Paper*. To appear in *Logic Colloquium '90*, Springer-Verlag.
- [15] Dov Gabbay. Theory of algorithmic proof. In S. Abramsky & D. Gabbay & T. Maibaum (eds.), *Handbook of Logic in Theoretical Computer Science. Volume 1*. Oxford University Press, 1991. forthcoming.
- [16] Dov Gabbay. Abduction and Analogy in Labelled Deductive Systems (Extended Abstract). Preprint, Dept of Computing, Imperial College, London. July 1991.
- [17] Dov Gabbay and Ruth Kempson. Labelled Abduction and Relevance Reasoning. In *Proceedings of the Workshop on Non-Standard Queries and Non-Standard Answers*, Toulouse, July 1991.
- [18] Dov Gabbay and Ruy de Queiroz. Extending the Curry-Howard interpretation to linear, relevant and other resource logics. Preliminary draft in [10]. Final version to appear in the *Journal of Symbolic Logic*.
- [19] Dov Gabbay and Ruy de Queiroz. An attempt at the functional interpretation of the modal necessity. Preprint (Draft), Dept of Computing, Imperial College. Presented at MEDLAR 18-month Workshop, Torino, Apr 27 - May 1, 1991.
- [20] Alfred Kobsa and Wolfgang Wahlster. *User Models in Dialog Systems*. series Symbolic Computation, Springer Verlag, 1989.
- [21] Hans Jürgen Ohlbach. A resolution calculus for modal logics. In *Proc. of 9th CADE, LNCS 310*, pages 500-516. Heidelberg, 1988. Springer Verlag. extended version: SEKI Report SR-88-08, FB Informatik, Univ. of Kaiserslautern, 1988.
- [22] Hans Jürgen Ohlbach. Semantics Based Translation Methods for Modal Logics. *Journal of Logic and Computation*, 1991. forthcoming.
- [23] Hans Jürgen Ohlbach and Andreas Herzig. Parameter structures for parameterized modal operators. In *Proc. of IJCAI 91*, San Mateo, California, 1991. Morgan Kaufmann.
- [24] Christoph Weidenbach and Hans Jürgen Ohlbach. A Resolution Calculus with Dynamic Sorts and Partial Functions. *Proc. of ECAI-90*, Stockholm, 688-693, 1990.
- [25] Larry Wos and Ross Overbeek and Ewing Lusk and J. Boyle. *Automated Reasoning. Introduction and Applications*. Prentice-Hall, Englewood Cliffs, 1984.

FOF PRODUCTION THEORY: TOWARDS AN INTEGRATED THEORY FOR ONE-OF-A-KIND PRODUCTION

J.C. Wortmann
Eindhoven University of Technology
Department of Industrial Engineering and Management Science
P.O. Box 513
NL - 5600 MB Eindhoven

SUMMARY

The Basic Research action 3143 "FOF PRODUCTION THEORY" aims at integration of various scientific disciplines involved in the design of production systems. The project is limited to one-of-a-kind production, and the project is especially interested in computer integrated manufacturing (CIM).

The central idea is, that the alternatives available in redesigning a factory cross the boundaries of established scientific or engineering disciplines. Therefore, these alternatives or "design choices" play a major role in our approach. Similarly, the performance indicators which measure the suitability of a redesign, cross the boundaries of established disciplines. Consequently, the FOF project focusses on these performance indicators as well.

The relationship between design choices and performance indicators is modelled in a qualitative manner by a so-called connectance model. This model is implemented in a tool called REMBRANDT, which allows us to browse through available design theory. Detailed knowledge of relations between design choices and performance indicators is being developed in a number of so-called drawing models. These drawing models are quantitative in nature, and can be fed with data from actual companies.

1. One-of-a-Kind Production and CIM

The markets for consumer goods are characterized nowadays by an increase of variety, while at the same time showing steadily decreasing product life-cycles. Improved delivery performance for short and unpredictable life-cycles is only possible with small batches. In addition, tailoring the product to the individual customer's needs is increasingly important in quality improvement. This tendency also results in production in small batches, which are often driven by customer orders. Ultimately, this leads to one-of-a-kind production.

Traditionally, one-of-a-kind production (OKP) has been associated with capital goods. However, European producers have shown a poor delivery performance, which resulted in a loss of market share in branches of industry such as shipbuilding (see Burbidge (1)). Consequently, the general public does not associate OKP with the future of European industry, but with its history. Hopefully, this is a misunderstanding of the general public.

The fact that consumer goods production itself moves towards OKP is only half of the story. In addition, the innovation and redesign of the production process is in many plants a considerable activity. It is not uncommon nowadays, to find a repetitive manufacturing production site employing 10%-15% of its direct labour in an OKP

machine shop involved in redesigning the repetitive factory. No doubt, the real issue in such plants is the performance of such an OKP-system.

There seems to be a common agreement that future production systems should be more "integrated" than the level of integration which is obtainable by now. Unfortunately, there is much less agreement about the meaning of the term "integration". This term is often used in connection with Computer Integrated Manufacturing, CIM. For proper understanding of integration of scientific disciplines aiming at the design of production systems, it makes sense to consider the term CIM first.

Figure 1 shows the CIM-architecture which is central in many papers about CIM. In this picture, the three bubbles at the ridge of the circle have something in common: these bubbles represent established business processes in most organizations. The bubble at the heart symbolizes integration, but it is different from the other three bubbles: this bubble merely suggests that, if the business processes are supported by information technology, these processes can exchange messages between computerized subsystems. In other words, the business processes are all connected to each other by some way of automated communication.

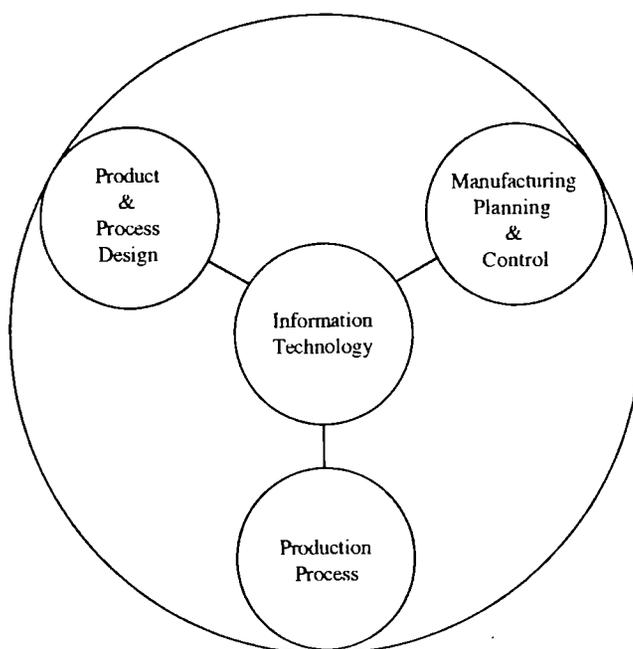


Figure 1. CIM-architecture (derived from Gunn (2)).

The main direction for improvement of today's factories lies in improved interaction between business processes. Each business process should give due recognition to the impact of its own actions on other business processes. Techniques like "design for production" or "design for logistics" are important examples for illustrating the need for interaction between business processes. This may be paraphrased as "Human Integrated Manufacturing". It is not our intention to deny the importance of information technology here. On the contrary, we believe that information technology enables us to obtain a much higher degree of interaction between business processes. However, the goal is not to have a fully automated factory, but to have full support of communi-

cation between interacting business processes. We shall use the term "integrated manufacturing system" to denote interacting business processes, giving due recognition to the impact of their actions for the performance of other business processes.

Now consider the design of such an integrated production system. Several questions emerge immediately.

Firstly, what is the object, i.e., what is being designed? Is it the plant layout or the local area network? Is it the decision-making structure or the salary system?

Secondly, what are the objectives of a design? In other words, which perceived performance criteria play a role in such a design activity?

Thirdly, how do we know, that a particular design will yield a particular performance? In other words, what is the scientific status of our design activities?

Fourthly, how can we design a series of small steps in redesigning an existing factory in a desired direction?

Fifthly, who is the designer? Is it an outside person, making clean rational calculations, or is it the organization itself, who is involved in redesign?

To help answer the preceding questions, the FOF project aims at developing a designer's workbench. In an integrated design, it is necessary to organize our knowledge. This means, that the objects of design, the performance criteria, and relationships known from different scientific disciplines has to be brought together into an organized set of models. This is, in essence, our idea about integration. The workbench will allow experimenting with a number of design choices to test, design or redesign a manufacturing system. The effect changes in design choices will have, is measured on a set of performance indicators.

2. Typology of OKP

In many cases it is convenient to regard one of a kind production as consisting of a number of operations. At an aggregate level, the following operations might typically apply:

- Design
- Process planning
- Component production
- Assembly

The two first are frequently referred to as engineering, and the two latter as production. Engineering and production comprise both processes of major interest.

Including the management type of operations, the manufacturing company can be regarded as consisting of three interdependent processes. This is illustrated in Figure 2. A process is defined as a set of related operations performed on, or in connection with, a flow of concrete or abstract items. In Figure 2 the processes are:

- Production
- Engineering
- Management

Production is connected to a flow of materials. The purpose of production is to transform raw materials into finished products.

Engineering is connected to a flow of technical information, usually represented as drawings or other documents. The purpose of engineering is to provide technical specifications on what products to produce and how to produce the products.

Management is connected to a flow of operational information, usually represented in the form of work orders or planning or status documents. The purpose of management is to release and monitor work orders for production and engineering.

In addition to the information shown in Figure 2, any system will also be exposed to external input, such as f.ex. product demand.

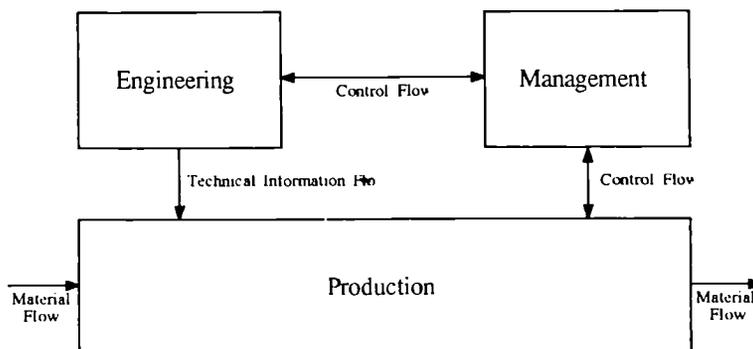


Figure 2. Processes in an OKP System.

A common distinction of different types of supply systems to the market is the distinction between standard-products supply and customer-order driven supply. However, customer-order driven supply encloses a number of different situations. For example, shipbuilders, maintenance shops, construction companies, and automotive component subcontractors may all be considered as customer-order driven suppliers. These production systems are so different, that a typology is needed.

Basically, a typology may be derived by following a customer order through an OKP factory. At an aggregate level the operations performed are as mentioned above (design, process planning, component production, assembly). In principle, each of these operations may be based on a standard solution of a customer order. This will give 24 combinations. The combinations may in practice be larger, since the mentioned operations may be of different type for each product or even component.

The complexity may be even larger, taking into account that engineering fall into different categories dependent on what is selected as input basis:

- Engineer from scratch
- Engineer from components
- Engineer from product

For this report is sufficient to review a simplified classification, limiting strongly the number of alternatives, as developed by Wortmann (3). This paper distinguishes the following two questions in order to create a typology (the typology is shown with examples in Figure 3):

- A. Which activities in the primary process are customer-order driven?
- B. Which investments (in e.g., product-design, resources, procedures, or supporting activities) are customer-order independent?

Based on question A, a well-known dimension emerges, viz.:

- A1. Make to stock: only distribution is customer-order driven
- A2. Assemble to order: assembly and distribution are customer-order driven
- A3. Make to order: purchasing, component manufacturing, assembly and distribution are customer-order driven
- A4. Engineer to order: even (part of the) product design is customer-order driven

Based on question B, a much less known dimension emerges, viz.:

- B1. Product-oriented systems: these systems supply the market with products which have been designed (to some extent) independently of existing customer orders

B2. Capability-oriented systems: these systems offer particular skills or resources, but not predefined products, to the market.

	A4	A3	A2	A1
B1	Airplanes; Packaging machines	Vessel engines; Standard professional equipment	Trucks; Computer systems	Furniture; Consumer electronics
B2	Software development; Civil engineering	Maintenance shop; Foundry; Forge shop	Building construction work	Supply of car outlets

Figure 3. Typology of one of a kind production (examples).

As always with typologies of companies, many companies do not fit neatly into this scheme. Usually, this is because a company may produce different product families or operate on different markets. This leads to different positions in Figure 3. However, Figure 3 illustrates the variety of situations which might be covered by the term "one-of-a-kind".

3. Currently Available Knowledge on OKP-Systems

In the remainder of this report, we will concentrate on column A4 and row B1 of Figure 3, unless explicitly stated otherwise. The last decade has shown an affluent wave of literature on engineering, production and production management. Unfortunately, nearly all material is rooted in the production of standard products (row B1, columns A1 and A2 of Figure 2). As argued in Bertrand et al (5), a production control system such as MRP II is based on standard products, produced in (large) batches. The OPT system is even more based on column A1 of Figure 2.

A production system design philosophy such as Just-in-Time (JIT) production, originates from automotive industry. It carries many elements which are relevant for line-assembly only. The same holds for Total Quality Control (TQC). Anyone who tries to apply these concepts to engineering design work is struck by the implicit assumption of standard products. This is true even for a book which takes distance from technicalities and details (such as Garvin (6)).

Computer Integrated Manufacturing (CIM) is another field where claims of applicability to OKP are suspect. Gunn (2), for example, stresses the fact that CIM has to go together with TQC and JIT. We will argue that information technology can contribute considerably to OKP-systems' performance, but not in the same way as elsewhere.

Theories on product design (such as Wheelwright and Sasser (7)) are typically focusing on customer-order independent design (for an overview, see Sederholm (8); the general nature of design is discussed in Takala (9)). Although this is certainly valuable, it is not dealing with one of a kind production. Here, again, there seems to be a lack of interest in current literature.

A notable exception is the literature on Group Technology. Burbidge (1) is focusing explicitly on engineering companies. However, the ideas of Group Technology are mainly described for component manufacturing. In fact, the close connection of Group Technology to Period Batch Control, advocated by Burbidge, suggests that the majority of the parts produced are standard parts. However, many ideas from Group Technology are formulated in quite a general way. Therefore, the attempt to specify Group Technology for OKP seems worth while.

It is evident that full scale Just In Time or Optimized Production Technology is inappropriate for OKP systems. However, it is possible to apply some major elements of JIT (such as elimination of waste, in-time deliveries over the total logistic chain, set-up time reduction, good control over the process, co-makship, Total Quality Control, multiskilled personnel, internal standardization, information systems to manage changes, preventive maintenance) successfully in OKP systems.

The same observation applies also with OPT. The fundamental idea to make a distinction with the activation of a resource vs. the beneficial use of a resource is valid in any type of a manufacturing system. In designing production management systems also the bottleneck approach is appropriate and can thereby simplify production management of an OKP system.

There is another wide misunderstanding of the potential of CIM technology in OKP industry. According to this CIM investments are oriented towards repetitive manufacturing, exclusively. However, without CAD / CAM the OKP companies are doomed to be eternally blamed for long throughput times, high overall costs, poor quality, slow reaction and long and poor delivery performance. Fortunately, there are some symptoms that this misunderstanding is losing its pace.

When considering theories of production organization, such as JIT, GT, or socio-technical design, there seems to be at least three "paradigms" or views which have to be synthesized. Each of these views provides ways to describe an existing or hypothesized production system. Each view relates design alternatives to performance indicators. Therefore, each view presents an evaluation of an existing production system with respect to particular performance indicators. These views are:

- A framework with theories about the appropriate way to structure the workflow through a factory. This workflow is not restricted to "physical" transformation of material. These theories lead to a structure of the workflow which is closely connected to an organizational structure in terms of departments, groups, task forces, teams, etc. (This is the workflow view mentioned in chapter 1)
- A framework with theories about the internal structure of the resources. In component manufacturing, this structure consists of the physical layout, the equipment, the task structure of individuals and groups. In OKP, and especially in customer-order driven engineering, the human aspect seems to be most important.(This is the resource view mentioned in chapter 1)
- A framework with theories about decision making. In OKP it seems that the boundaries between decision making and other activities (such as design) are less strict than elsewhere. (This is the organizational / decisional view mentioned in chapter 1)

A comparison of OKP and repetitive manufacturing is done in Table 1 (Falster (11)):

Manufacturing	Management	Engineering	Info Tech		Resource	Organisation
Repetitive	JIT		Workst NC	CAD CAM	Machine	Functional / Line / Groups
One-of-a-kind		CIT	Comm	CIM	Human	Groups

JIT = Just In Time

CIT = Concurrent in Time, e.g. Concurrent Engineering

Table 1. Comparison of key elements in One-of-a-Kind and Repetitive Manufacturing.

4. Towards an Integrated Theory

Redesign is an interactive pattern of activities where the scope and focus change frequently. This means that OKP design choices should be tackled at different layers. The three levels adopted in our basic model are the following:

- 1 Strategy
- 2 Structure
- 3 Operations.

The highest layer (strategy) deals with the strategies to be adopted and the inherent competition factors. It also deals with the economics (particularly long term) of the manufacturing systems. Typical design choices to be made at this layer are listed below (cmp. e.g. Hayes and Wheelwright, 1985 (28)) :

- market niche (customer specified / modified vs. standard; brand vs. bulk; cost minimization vs. differentiation)
- product technology (mechanics, electronics, software,)
- process technology (scale, flexibility, dedication)
- facilities (location, size, specialization)
- integration (vertical, horizontal, extent, balance)
- suppliers (number, structure, relations, make / buy)
- human resources (selection, training, compensation, security)
- quality (image, responsibilities).

The intermediate layer (structure) deals with the production system as such, i.e. its components (primitive system) and the relationships between them (restrictions). Some typical design choices at the structural layer are

- products
- resources
- manufacturing layout
- organization

The layer of operations deals with decisions how the inputs of the manufacturing system are applied to the structural manufacturing system. Some typical design choices at the operational layer are:

- operative and medium term decision rules and criteria
- operative and medium term management functions.

The operational input fall into two categories:

- demand pattern
- supply pattern.

The demand pattern is represented as the customer order flow while the supply pattern is resulted from the availability of subcontractors, material suppliers and the resources of the business unit itself.

In any layer of the total conceptual model some of the design choices remain fixed, while free choice can be exercised over the others. There are also situational conditions (such as demand pattern) more or less given at each layer.

Because of the hierarchy of the layers there are most degrees of freedom at the highest layer (strategy), because both structural and operational design choices are at least in principles open. On the contrary, when making operational design choices both the strategic and structural layers are more or less fixed.

Fixed design choices and situational conditions are together called the frame conditions in our total model.

5. DC-PI Network

When the designer takes a large scope, (s)he tends to be more focused on strategic management issues. At strategic level, a designer with a wide scope needs to navigate carefully through theoretical cause-and-effect chains before selecting a particular promising area, suitable for a close up.

In the remainder of this chapter, it will become clear, that such a means of navigation has been developed. It connects design choices (DCs) to performance indicators (PIs), and it will be called therefore the DCPI network (see chapter 14). It has the form of a so-called connectance model (Burbidge, 1984 (17), Eloranta, 1981 (29), Karni, 1990 (30)). The theoretical background of connectance models is in the soft systems science (e.g. Checkland, 1985, (31)), which has been applied, e.g. on the research of the future, in qualitative economics as well as in qualitative simulation. The type of relationships implemented in the FOF model are the following:

- Affects (is affected by)
- Is element of (has element)
- Increases (is increased by)
- Decreases (is decreased by)

Because of the variety of the permitted relationships, the mathematical properties and thereby the reasonable operations in the connectance model are scarce. Transitivity, e.g., is limited to just a few types of relationships.

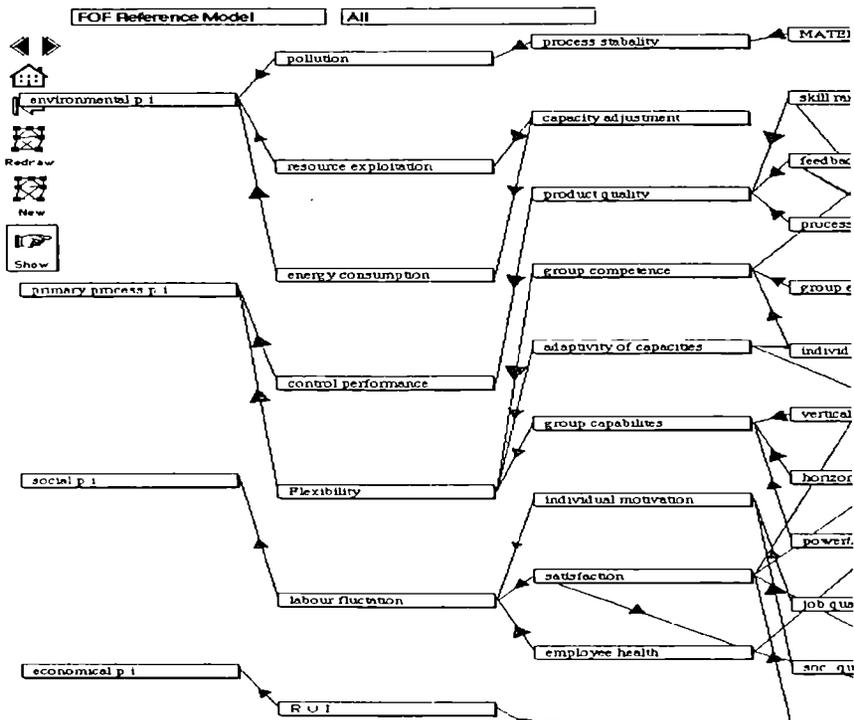


Figure 4. Sample from the DC-PI network in REMBRANDT.

At the structural or the operational layer, when the designer takes a more narrow scope, there emerges a need for more precise modeling of relationships between

design choices and performance indicators, eventually based on situational factors. These more precise models are called relationship models. Relationship models are often quantitative models. Such quantitative models for parts of factories are usually implemented as simulation models. These models are still relating design choices to performance indicators, but they do not claim to be comprehensive.

Simulation models and other relationship models can be fed with actual descriptive data about the factory to be redesigned. Therefore, relationship models should be distinguished in:

- Reference relationship models (general theories and knowledge)
- Particular relationship models (company selected knowledge, data and facts)

Finally, there is one advantage of the architecture above which is worthwhile to notify. In principle, the structure of the DCPI network and the various drawing models is such that it can be maintained if our knowledge about the (OKP) manufacturing systems grow. More explicitly, the DCPI network is represented in a tool called REMBRANDT. Part of this network is presented in Figure 4 above.

6. The FOF design framework

The design framework is based on two pillars (Falster (11)):

1. Descriptive production theory which originated in resources/workflow modelling, the so-called Walrasian production model and its generalization to dynamic modelling of control/decisions/organization based on control theory and the GRAI Grid.
2. Theory of factory (re)design (Takala (15)).

This framework exists at two different levels:

A - The level of reference models

B - The level of particular models.

This split in reference and particular models are in line with the ESPRIT CIM-OSA model (26). Reference models represent available theory. These models link general design choices via intermediate variables to general performance indicators. They may give due recognition to the existence of situational factors, but these models claim to be generalizations over such situational factors. Particular models represent an abstract example of a particular case or an existing or conceived real-life situation. These models show particular design choices which have been made in a particular case. They enable the designer to specify alternatives for these design choices, and to compute the consequences of these alternatives within a limited domain of knowledge.

The reference model can be split on two sub-levels, A1 and A2. In the theoretical framework these levels represent:

A1 - The primitive system, i.e., the various components of the system.

A2 - The constraints, i.e., how the system components are connected together (topology of the system).

In the design framework the counterpart to the primitive system is denoted an entity model. An entity model thus represents a description scheme for a particular real-life situation by design choices and performance indicators (Falster (11)). The counterpart to the constraints is called a relationship model (see section 5). The relationship model has several layers of detail. The top layer is constituted by the DCPI network. Links in the network can be replaced by more detailed, quantitative models, in an open architecture. The relationship model represents a heuristic to interconnect design choices, performance indicators and intermediate/independent variables of the entity model (Falster (11)).

In summary, the design framework includes, as indicated in figure 5:

- Design reference model consisting of an entity model and a relationship model
- Relationship models consist of more detailed, quantitative reference relationship models
- Particular models, based on reference relationship models fed with actual data

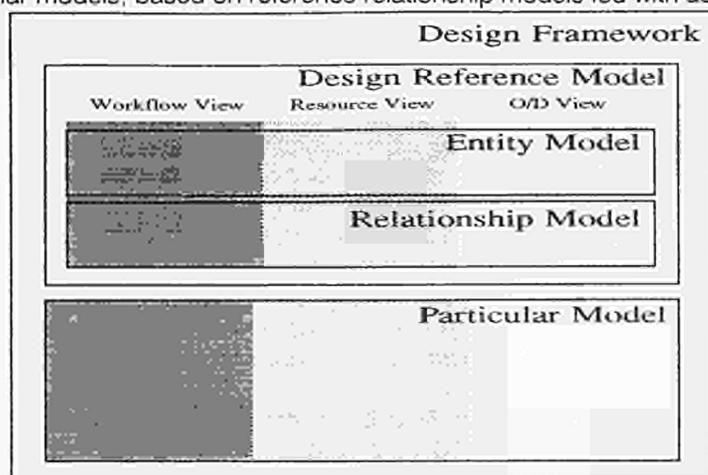


Figure 5 Models and views in the design framework.

In figure 6 the design reference model is explained.

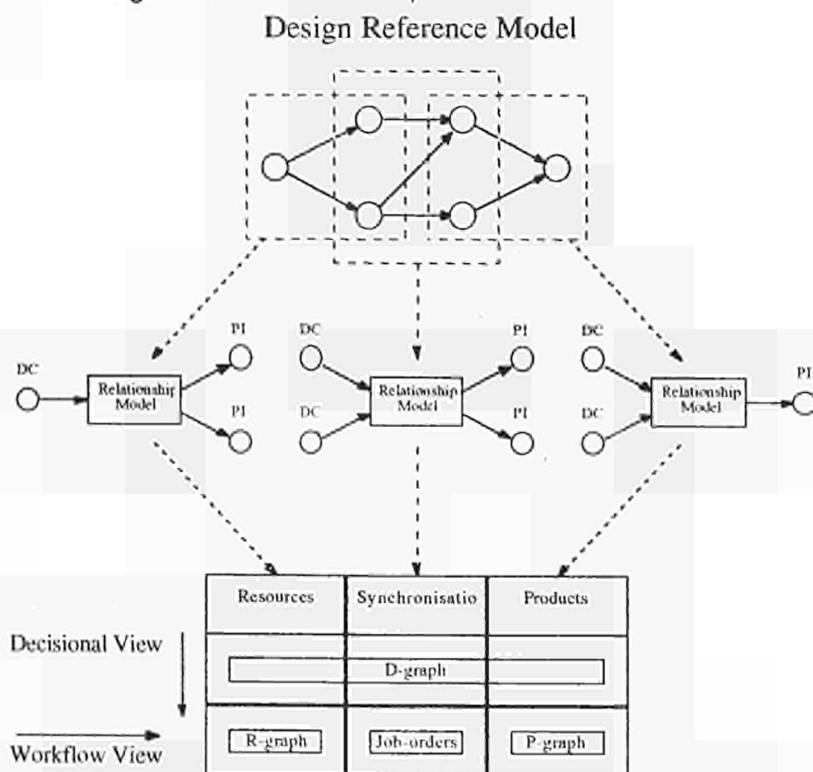


Figure 6. Views and basic data structures in a Design Reference Model.

All models may be seen from any of the three views:

- Workflow
- Resource
- Organizational/decisional

We are now in a position, to explain more about the detailed relationship models. Such a relationship model contains limited data (PIs and DCs), and it models only a subset or a limited part of the total OKP system. Relationship models may comprise a view, part of a view, or may even contain parts of several views. A relationship model can be used as a template to describe a particular piece of reality in a quantitative way. We distinguish between reference relationship models and particular relationship models. The difference is that a reference relationship model will operate on variables, while the particular relationship model is fed with values of these variables.

In the top part of the figure the design reference model is shown. In the middle part the design reference model is sliced into views and an entity model (DC, PI) and a relationship model (RM) is shown for each view. In the lower part the views connectance to the basic data structures are shown as well as vertical decomposition of the GRAI grid.

The aim of the workflow view is to describe the primary flow, grouping of resources and products/work. These may be based on group technology (factory flow analysis and departmental flow analysis). The aim of the resource view is to describe the internal structure of teams of humans. The aim of the decisional view is to describe the decisional flow and logic in a multilevel hierarchy.

The FOF project is dealing with both the engineering and production processes within the OKP-system. We consider engineering processes as similar to production processes. In other words, engineering work produces output which is planned and delivered in the same way as materials are.

The output of this engineering work, "paperwork", documents, or better: information, can be managed according to known principles in production. The information-producing nature of engineering output, will enable the use of IT in managing the timing of the output. This sheds a new light on CIM in OKP.

7. Simulation models

The components of a design reference model are depicted in figure 7. The idea is that design choices (DC) are changed. The effect on the OKP system is measured by the performance indicators (PI). The DCPI network referred to in section 5, defines the relationship between DCs and PIs.

The design choices are made about (f.ex.) physical layout, sociotechnical structure, functional structure, decision hierarchy and responsibility, or organisation. Design alternatives could be functional, process, or group layout, hierarchical or autonomous group structure etc. The aim of design choices is getting a better factory.

Better is measured in terms of the performance indicators that are deemed relevant for this decision. Performance indicators could be efficiency, quality of working life, decision stability, etc.

A relationship exists between each design choice and its relevant performance indicators. These relationships, obtained from theory or empiry, are expressed in models (relationship models). It is the task of the modeler, to specify these relationship models, and with that, to indicate the relationship between a design choice and its performance indicators.

In designing/redesigning or testing a manufacturing or production system while using relationship models, some information is given as input. This is basically the products to be delivered and the resources that the company have available. Both may be inadequately, insufficiently or only partly defined. They may all be changed or modified in the process of designing or redesigning the manufacturing system.

Variables

Models

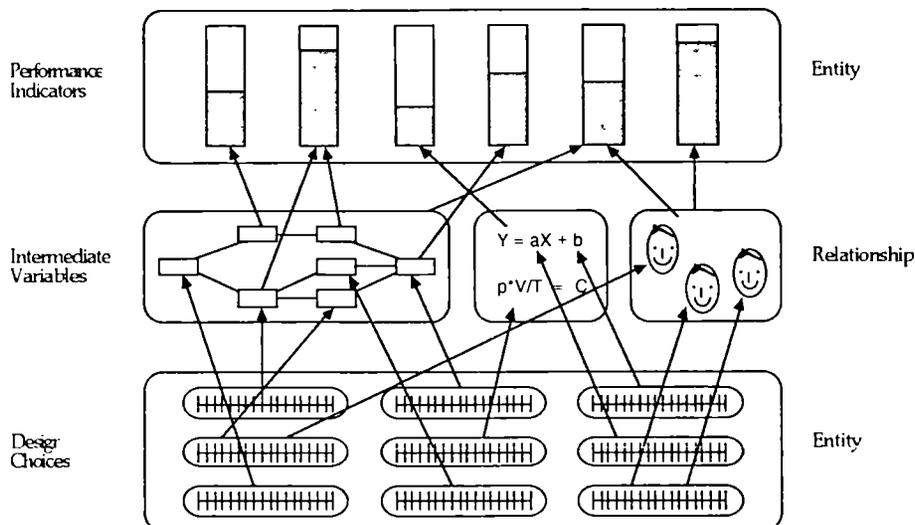


Figure 7. Components of a Design Reference Model.

The company exists to satisfy a requirement for products from its customers. This demand pattern is also assumed to be one of the basic input data of relationship models describing a part of the manufacturing system.

Finally, the company may operate different working hours on different resources. The operating hours is another basic input data, of the same type as demand.

All these input data can be categorized into either design choices or experimental input.

The design choices define a model of the production system as such, i.e., its components (primitive system) and relationships between them (restrictions). Some typical structural design choices are:

- Products
- Resources
- Production layout
- Resource organization
- Decision rules and criteria
- Management functions

The experimental input defines the input applied to a model of the production system. This input falls in two categories:

- Demand pattern
- Supply pattern

The demand pattern is the customer orders and the supply pattern defines the availability of subcontractors, suppliers and the working hours of the company's resources.

In designing/redesigning the system while using a relationship model, the various structural design choices are changed to demonstrate different performance. The performance is measured by performance indicators. In testing the system the structural design choices are kept fixed, while the operational design choices are changed and applied to measure different performance. Testing is to check how the structural system responds to different operational conditions.

In any application of the model some design choices and input parameters remain fixed, while others are varied. The set of fixed design choices and input parameters for a specific application is referred to as the frame conditions under which the model is used.

In summary, the use of the model is visualized in Figure 8 (Rolstadås (4)).

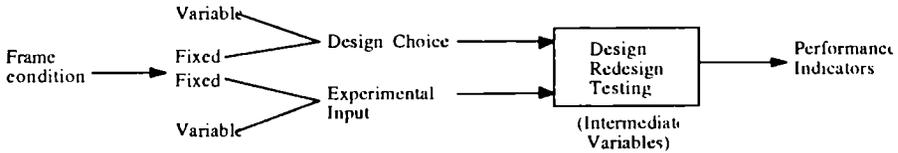


Figure 8. Input/output in Operation of a Relationship Model of an OKP System.

There may be only one relationship model built into one simulation model, and there may be many relationship models built into one simulation model.

It is preferred to implement several relationship models into a single simulation model. This makes programming and execution more efficient, and insures that the characteristics of the same production systems are the basis of many evaluations. It is not possible, however, to achieve the implementation of all relations into a single simulation model.

There are three types of relationship models: Analytic, dynamic, and event based. These types of models must be implemented in three different types of simulation models and tools.

Models differ in the method of calculation that is required to evaluate the relationships between Design Choices (DCs) and Performance Indicators (PIs). From this viewpoint there are three types of relationships, and therefore three types of required simulation tools:

- The relationships between some DCs and PIs can be evaluated with analytic models. Tools for the evaluation of this type of relationships could be spreadsheets or other single shot calculation tools.
- The relationships between some DCs and PIs can only be evaluated with continuous dynamic models. This is the case when there are feedbacks, and time dependent effects. Tools for the evaluation of this type of relationships could be Stella, Dynamo, or other multi period evaluations models.
- The relationships between some DCs and PIs can only be evaluated with discrete event models. This is the case when characteristics of the individual events have an impact on the behaviour of the system, and need to be taken into account during evaluation. Time dependent effects also play a role. Tools for the evaluation of this type of relationships could be GPSS, Taylor, Exspect, or other event driven simulation tools.

Characteristics of the models that make the use of several simulation models per tool necessary are the time scales on which choices are made, and the impact of DCs on PIs takes effect. It is not useful to evaluate effects that take place on different time scales in the same model.

8. CONCLUSIONS

The value of the FOF conceptual model is that performance indicators from various fields of study can be related to the same design choice. This allows an interdisciplinary, multi-criteria evaluation of a design choice.

There may be relationship models that connect only one design choice with one performance indicator, there may be relationship models that connect a single design choice with several performance criteria, and there may be relationship models that connect many design choices with many performance indicators. A single design choice may influence one or many models. A single performance indicator may appear in one or many models.

The simulation models are implementations of the relationship models in some executable tool, such that design alternatives can be evaluated.

The FOF aim of making an IT based designer's workbench requires that the relationship models can be used in computers. To this end, the relationship models are built into simulation models. These simulation models can then be executed to evaluate design alternatives.

The various simulation models will be provided as a collection of software systems designed within the project. The main purpose of this software will be to provide a demonstrator that will demonstrate the achievements of our project towards an integrated theory.

Acknowledgements

This work is based on many contributions of nearly anyone involved in the FOF project. However, the major part is derived from the final report of workpackage 2. This report was mainly prepared by Peter Falster and Asbjørn Rolstadås. The final version would not have been possible without the help of Rob Kwikkers and Pierre Breuls.

References

- (1) Burbidge, J.L.: Group Technology in the Engineering Industry. (Mechanical Engineering Publications LTD, London, 1979).
- (2) Gunn, T.G.: Manufacturing for Competitive Advantage. (Ballinger/Harper and Row, Cambridge, Mass., 1987).
- (3) Wortmann, J.C.: Towards and Integrated Theory for Design, Production and Production Management of Complex, One of a Kind Products in Factory of the Future. In: Commission of the European Communities (ed.), ESPRIT'89, Proc. 6th Annual Esprit Conf. (Kluwer Academic Publishers, Dordrecht, 1989) pp. 1089-1099.
- (4) Rolstadås, A.: A Conceptual Reference Model Seen from the Functional View. Esprit Basic Research Action no 3143, FOF/SINTEF/2-16, Aug 1990.
- (5) Bertrand, J.W.M., Wortmann, J.C., and Wijngaard, J.: Production Control - A Structural and Design-Oriented Approach (Elsevier, Amsterdam, 1990).
- (6) Garvin, D.A.: Managing Quality.
- (7) Wheelwright, S.C. and Sasser, W.E., Jr.: The New Product Development Map. (Harvard Business Review, May-June 1989).
- (8) Sederholm, B.: Design Methods in Practice. (FOF ESPRIT BRA 3143, Helsinki University of Technology, Dec 1989).
- (9) Takala, T.: Design Theory for the Factory of the Future. (FOF ESPRIT BRA 3143, Helsinki University of Technology, Dec 1989).

- (10) Falster, P.: The Conceptual Model and Related Topics. (FOF ESPRIT BRA 3143, Technical University of Denmark, Dec 1989).
- (11) Falster, P.: Towards a Conceptual Model. (FOR/DTH/WP2/3, Oct 1990).
- (12) Wortmann, H.: The Structure of the Final Results of FOF 1. Esprit Basic Reserach Action no 3143, July 1990.
- (13) Marcotte, F.: FOF: Organizational/Decisional View. FOF/GRAI/200, June 1990.
- (14) Pedersen, A.: Theoretical Framework for One of a Kind Production - from a Systems Science Point of View. Proposal for Ph.D. report. Electric Power Engineering Department, Technical University of Denmark, Lyngby, June 1990.
- (15) Takala, T.: Design Theory. (FOF ESPRIT BRA 3143, Helsinki University of Technology, 1990).
- (16) Wortmann, H.: Towards an Integrated Theory for Production. Esprit Basic Research Action no 3143, July 1990.
- (17) Burbidge, J.L.: Classification of Production System Variables. In Hubner (ed) Production Management Systems: Strategies and Tools for Design. Elsevier Science Publishers, IFIP 1984.
- (18) Skarpeid, H., Qvistgaard, T.: The Functional Approach to FOF, FOF/SINTEF/2-17, Aug 1990.
- (19) Doumeings, G.: Methode GRAI: Methode de Conception des Systemes de Production. These d'Etat; Automatique. Universite de Bordeaux 1 (Nov 1984).
- (20) van der Heiden, G.: PITF report on Design Choices and Performance Indicators. FOR/PITF/TUE 1990.
- (21) Skarpeid, H.: Consolidated Report Functional View. SINTEF, July 1990.
- (22) Schalla, A.: The Reference Model for Decision Making. BIBA, Bremen, May 1990.
- (23) Rolstadås, A.: Structuring Production Planning Systems for Computer Applications. APMS 1987, Elsevier, 1987.
- (24) Kwikkers, R.: The Role of Simulation Models in the Conceptual Model. FOR/MGMT/173, June 26 1990.
- (25) Wortmann, H.: Towards One of a Kind Produciton: The Future of European Industry. ESPRIT Project 3143. APMS'90, Helsinki, Aug 1990.
- (26) ESPRIT Concorium AMICE (Eds): Open System Architecture for CIM. Springer-Verlag 1989.
- (27) Burbidge, J.: Production Flow Analysis. Oxford University Press, 1990.
- (28) Hayes, R., Wheelwright, S.: Restoring our Competitive Edge. Wiley, 1985.
- (29) Eeloranta, E.: An Approach for Gross Design of Operations Management Systems. Ph.D. Thesis, Helsinki Univ. of Tech., 1981.
- (30) Karni, R., Gal-Tsur, A.: Frame-based Architectures for Manufacturing Planning and Control. Working Paper, Technion - Israel Institute of Technology, Haifa, 1990.
- (31) Checkland, P.: Achieving "Desirable and Feasible" Change: An Application of Soft Systems Methodology. J. Opl. Res. Soc., Vol. 36, No 9, pp 821 ... 831, 1985.

Turning the Formal Verification of VLSI Hardware into Reality¹

L. Claesen², D. Borrione³, H. Eweking⁴, J.L. Paillet⁵, P. Prinetto⁶

Abstract.

This paper presents an overview of the different aspects in the area of the formal verification of VLSI hardware. For particular aspects of the problem area the adequate approaches are being addressed. In this respect an overview of major directions and achievements in the area of formal hardware verification as under research in the CHARME ESPRIT Basic Research Action are presented. All partners are convinced that formal verification, given the appropriate methodologies, algorithms and formalisms, will find its place in actual CAD systems for industrial hardware designs. Research results include among others a link-up of formal verification tools to VHDL as well as the demonstrated Mi formal verification of actual VLSI chips of over 32000 transistors from the layout up to high level algorithmic specifications.

1. Goals of Formal Hardware Verification: Why is it needed?

The constant evolution in the microelectronics technology continuously allows to integrate larger and larger systems in integrated circuits and systems. This has its consequences in the fact that complicated chips of over 1 million transistors are realizable and that semi-custom approaches of standard cells and gate arrays have emerged to an enabling technology, not only for the classical electronics systems industries but also for innovative SME's. Electronics systems emerge in all aspects of every day life such as consumer electronics, telecommunication, HDTV, speech and image processing, computing systems, automotive applications etc... Instead of being *technology limited*, complex electronic systems have become *design limited*. It is indeed crucial that these complex systems, as required by industries incorporating electronics in their products, are *first time right*. Design errors should be detected as soon as possible in the design phase, because erroneous designs will introduce considerable delays in the introduction of innovative products on the market. These on their turn induce unacceptable losses in profit.

The classical way in which digital systems are being evaluated for design correctness is by a huge amount of simulation experiments. It is however well known that, except for trivial and obvious examples, exhaustive simulation covering all possible patterns is impossible to perform due to the problem of combinatorial explosion.

Formal design and verification techniques [4] attempt to address the design problem in an analytic way in order to obtain mathematical guaranteed correctness with respect to the modeling method used. Automatic synthesis from high level specifications [1,21

1 Research sponsored by the EEC under the ESPRIT Basic Research Action 3216: CHARME

2 IMEC / Kath. Univ. Leuven, Kapeldreef 75, B-3001 Leuven Belgium

3 Lab. IMAG-ARTEMIS, Inst. B.P. 53X, F-38402 Grenoble Cedex, France

4 Inst. fuer Datentechn., Merckstrasse 25, Techn. Univ. Darmstadt, D-6100 Darsmtadt, F.R.G.

5 Université de Provence, CASE Y, 3, Place Victor Hugo, 13331 Marseille Cedex 3, France

6 Polit. di Torino, Dip. di Autom. e Inf., Corso Duca degli Abruzzi, 24, I-10129 Turin, Italy

is an approach that can already provide some solutions for specific design aspects and abstraction levels. However, due to the unbeatable human insight in the underlying design problems, there will always be a large human and manual contribution in the design of complex digital systems, which need to be verified. Even in the case of so called "correctness-by-construction" methods, formal verification methods will enable the discovery of software bugs in the synthesis systems at hand by cross checking the results of the synthesis process.

Formal methods have already a long tradition and constitute a number of aspects and different approaches in the area of software development. In fact a lot of European research in this area is sponsored in ESPRIT projects [3]. The correctness of hardware designs have much more impact on direct costs involved in iteration cycles (due to VLSI processing etc.) and late introduction of products into the market. This motivates even more the need for formal verification in hardware than it is already in software design. The ongoing research in hardware verification, as is the goal in the CHARME project, is driven by a need to verify complex digital systems in as automatic a way as possible. This is required to foster acceptance in the electronic CAD community.

The goal of the ESPRIT CHARME Basic Research Action is to investigate, evaluate and prototype promising approaches that can help guarantee digital hardware correctness. For this purpose a grouping of researchers with backgrounds in electrical engineering, computer science and mathematics has been formed. These different backgrounds allow to put a number of different approaches and view points together in order to develop appropriate techniques and methodologies in order to address the problem of hardware correctness in the best way. All partners in the action are convinced that formal hardware verification can be converted into actual CAD systems and design practice. Therefore prototyping and application of the ideas to actual electronic design problems is ultimately important. Only the head lines of the research in CHARME are described in this paper.

Formal methods address a large number of design areas and levels of abstraction and encompass several different approaches to tackle these problems. Therefore an overview of the major *design aspects and levels of abstraction* is given in section 2. Based on this design aspect and abstraction level classification a description of individual formal verification techniques addressing specific areas is given in section 3. In order to facilitate the use of specific verification techniques, *methodologies* and/or *design rules* could result in "Design for Verifiability" in similar lines as what has been achieved by "Design for Testability" [6] as described in section 4. In section 5 the major ideas for the future direction of the research in formal hardware verification are indicated followed by the conclusions in section 6.

2. Design Aspects, Levels of Abstraction

Format Methods is a term that is used to cover several meanings, related to either different *aspects*, different *levels of design abstraction* as well as different underlying *formalisms* and *methodologies*. In order to clarify this situation a problem oriented overview of design aspects and abstraction levels is given in this section. This classification is important because the different aspects and levels of abstraction often have given rise to specific developments of methods and algorithms for formal verification. An overview according to formalisms is given in [4]

In VLSI hardware design, errors can be introduced at several places such as layout rule violations, circuitry etc... For synchronous digital VLSI designs the verification of the correct functionality can be subdivided in to the aspects of timing-, electrical- and

behavioral correctness verification. In this paper we mainly concentrate on the correct implementation versus specification behavioral verification.

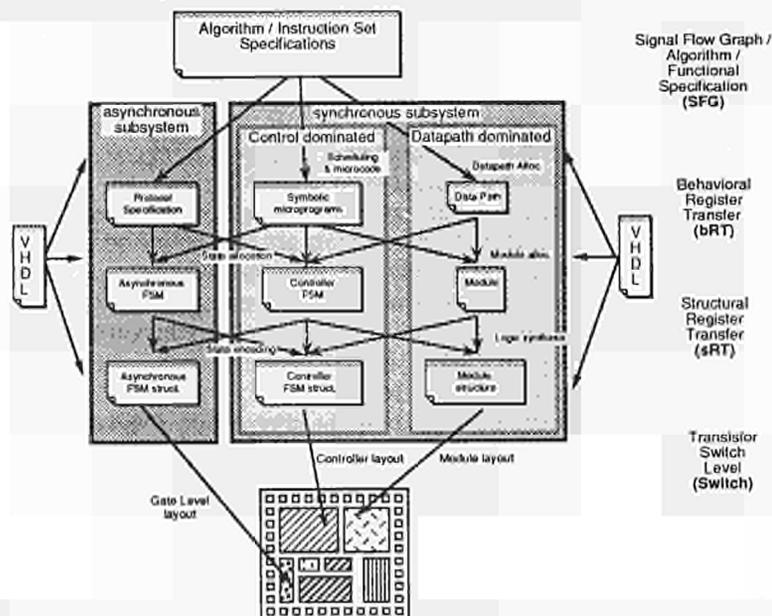


Figure 1: Aspects (horizontal) and Levels of Abstraction (vertical) in Hardware Design.

Figure 1 provides an overview of current digital system design, including the specific aspects when VLSI implementation is targeted. It is assumed that digital system design starts from a *formal* specification at the behavioral or signal flow graph level. Such specifications usually take the form of software models in classical programming languages such as FORTRAN or C. For digital system specifications at these high levels, several dedicated designer oriented system and hardware description languages as well as more mathematical languages have emerged. VHDL [5] is a standard system and hardware description language (HDL) that is gaining acceptance in industry and is supported by CAD vendors with compilers, simulators and logic synthesis systems. VHDL is a language that can not be overlooked anymore. It allows to describe digital systems at several levels of abstraction, using several description styles, namely the behavioral, structural and dataflow styles. Thus, the behavioral specification as well as the circuit implementation can be given in VHDL. For all these reasons, we believe that VHDL is a good candidate as input language for a Formal Proof environment.

2.1 Synchronous - Asynchronous Subsystems

In digital hardware systems, a major subdivision of *design aspects* is the subdivision of *asynchronous* and *synchronous* subsystems. To be able to manage the design process and for increasing testability [6], the largest part of current digital designs are synchronous.

2.1.1 Asynchronous Subsystems.

The interfaces to the externals of a system or other integrated circuits are most often realized by asynchronous subsystems. The asynchronous interfaces have to imple-

ment communication protocols among digital systems. They form the bridge between the asynchronous (interface) world and the synchronous regions on the chip. Such asynchronous circuitry usually consists of small parts of the global circuitry, but nevertheless have to implement the secure communication with the outside world.

2.1.2 Synchronous Subsystems.

For the synchronous subsystems a major subdivision of circuitry in *control dominated*- and *data path dominated* subsystems is possible (see fig. 1). The data path dominated subsystems usually implement the vector operations such as additions, multiplications, etc.. These data path subsystems are characterized by the fact that the circuitry is generated in relation to the chosen word lengths of the digital words being processed. In integrated circuits these are usually implemented using structured layout techniques. Typical applications of data path dominated circuits are for example ALU's, ACU's Multipliers etc... They are heavily employed in the data paths of computers as well as in digital signal processing applications [2].

Control dominated subsystems implement the control over the data path dominated subsystems as well as implement the digital control of an application at hand. Several of the current ASIC applications fall in this category. Control dominated systems are usually much more random in nature than data path dominated subsystems, and are usually implemented using random logic (currently often done as standard cells or gate arrays).

2.2 Levels of Design Abstraction.

An other classification of design aspects in fig. 1 is according to the levels of abstraction (in space and in time): (1) signal flow graph / algorithmic / instruction set level: **SFG**, (2) behavioral register transfer level: **bRT**, (3) structural register transfer level: **sRT**, (4) MOS transistor switch level: Switch, as described further on.

2.2.1 Signal Flow Graph / Algorithmic / Instruction Set Level.

At the highest level of design abstraction, the signal flow graph defining the behavior of the algorithm to be synthesized in hardware is considered. At the SFG level it is not specified *how* the algorithm is implemented in hardware. This could in fact be done in numerous ways (e.g. bit-serial, micro-code processor type of architecture (with several variants), bit-parallel or any combination of these). It is independent of how operations are performed, either on dedicated hardware blocks or on general purpose ALU's. It should be noticed, that very often the topology of the SFG will not directly correspond to the topology of the synthesized architectures.

In the case of processor design, the highest level of design abstraction is the specification of the instruction set (reference manual). This defines the processor-memory system as it is seen by the software programmer. The memory is seen as an array of addressable words of a given length, and only the internal registers of the processor which are made visible to the machine language are represented at this level. The internal structure of the processor, the processor-memory detailed exchanges, are not part of this level of specification. On the contrary, the emphasis is on the definition of the operation codes, addressing modes, and on the result of the execution of each individual instruction, all in symbolic form.

2.2.2 The Behavioral Register Transfer Levels (bRT-level).

In high level synthesis [1,2] the first steps consist of transforming the SFG level specification into a *data-path* and a *schedule*. In the behavioral register transfer level, the specific data path operators, such as the amount of ALU's, multipliers etc. and the bus interconnection, are known from the data path allocation. The allocation of variables to registers, and operations to data path operations over specific machine cycles is done in the scheduling. At the behavioral register transfer level, specific registers and data path hardware blocks are known. The micro-code to control the data paths, and the state encoding is in *symbolic form*.

2.2.3 The Structural Register Transfer Level (sRT-level).

At the structural register transfer level, the interconnection of logic building blocks, registers and state variables are modelled in the same way as they are implemented in the ultimate hardware. Starting from the schedule as available at the bRT-level, the controller is synthesized, starting from a generic structure. A specific state assignment has to be done. The data path hardware is realized by a library of parameterized building blocks. At sRT level, a logic description is available which corresponds to the structure and state encoding in the actual hardware implementation. In this paper, no distinction is made between the sRT-level representation and a gate level representation, as they are both very close together.

2.2.4 The MOS Transistor Switch Level (Switch-level).

When targeting towards MOS VLSI implementation of the system at hand, all digital circuitry has to be implemented in terms of MOS transistors. Transistors can be described in general at the circuit level using non-linear device models, that accurately model the voltage and current relationships at the terminals. For synchronous digital circuits, the switch level abstraction, that abstracts MOS transistors as switches and circuit nodes with a number of different strengths [7] has proven its usefulness.

3. The Formal Verification Map.

Given the outline of design aspects and levels of abstraction of fig. 1 as described in section 2, it is now better possible to describe the *Formal Verification Map* as indicated in figure 2. Here we indicate the most important developments and relate formal verification work (by the numbers in circles on figure 2 and near the titles in the text) and achievements in CHARME that address specific design aspects and levels of abstraction. This description starts from the lowest levels of abstraction as it is there that the most progress towards automated formal verification has been made.

3.1 Transistor Switch Level Analysis. (1)

After the generation of the mask layouts it is possible to extract the transistor networks as they are actually implemented on an integrated circuit. This is a good representation for the actual circuits and their behavior as they will ultimately be realized on silicon. In order to be able to capture all possible design errors (also in geometry and interconnections) this representation is a very good one to start the formal verification from. When abstracting to the behavioral modeling, the transistor switch level representation [7] is currently very well accepted to perform ultimate simulations

on in order to check the circuit functionality. Symbolic analysis techniques of switch level circuits [9,13] based on the solution of a number of systems of Boolean equations [8] have been developed. In the CHARME project similar techniques have been realized [11,12] in the BOTRYS program. An alternative approach based on the formulation of the transistor switch level model in predicate logic has also been worked out in CHARME [14] in the SWAN program.

These approaches have been used for the verification between the sRT and the switch level.

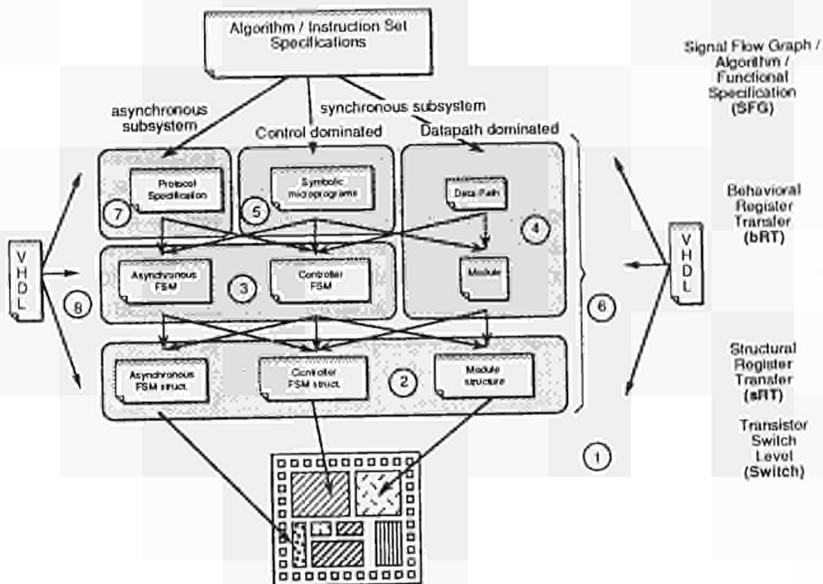


Figure 2: The Format Verification Map. The numbers in circles refer to descriptions of formal verification techniques in the text.

3.2 Combinatorial Logic Blocks. (2)

For the verification of different design representations at the sRT level, it is possible, due to the knowledge and use of the same state encoding of registers in specification and implementation to reduce the problem of formal verification to the comparison of the Boolean functions of the combinatorial logic blocks. For the comparison of Boolean functions, tautology checkers can be used. A comparison on the same benchmark set [17] of over 10 tautology checking algorithms, from all over the world including algorithms as developed by CHARME partners has been made. From this it has become clear that the methods based on the concept of OBDD's (Ordered Binary Decision Diagrams) [19] and their improved derivatives [20,21,22] are clearly superior to the other existing methods in their performance. The methods of OBDD's are currently recognized as the most efficient approaches for Boolean function comparison. OBDD's are used as the basic representation formalism of Boolean formulas within most of the research in the CHARME project [10,57,18]. The comparison of representations at the sRT levels ((structural) register transfer level & gate level) [20] has been used for

checking the correctness of *the individual combinatorial building blocks* in chips of up to 300,000 transistors [23] and have currently found industrial application in Bull.

For the verification of different representations at the sRT level in the assumption of the same state encoding, the LOVERT [57] program has been developed and coupled to VHDL. Implementation verifications of a 32-bit ALU and small microprocessors has been done in cpu seconds.

3.3 Finite State Machine Applications. (3)

The comparison of the Boolean functions of combinatorial blocks by means of tautology checkers (e.g. OBDD's) is only directly possible when the same state encoding is used in the two representations. In several applications the correspondance in the encoding of the two representations of digital circuits is not directly known. E.g. a controller can be defined at the bRT level with symbolic states and is implemented at a lower sRT level with specific states. To cope with this problem general algorithms for the comparison of two finite state machines, for which the state encoding and their correspondances is not known have been developed.

A major breakthrough in the complexities of comparisons of FSM's has been achieved with the symbolic representations (instead of specific enumeration) of the visited states [27]. The achievements in these comparisons have lead to verifications of FSM's with more than 1020 states, which corresponds roughly to some 60 state variables [29,30]. This means that small controler like circuits can be verified by this. Based on these symbolic techniques, a program called EPOS [65], for the verification of FSM's as black boxes has been developed in CHARME. In addition specialized algorithms [25,26] for FSM verification on the product machine have been developed in CHARME. The two algorithms differ in the following points: algorithm #1: forward time processing and explicit enumeration to dynamically build the product machine; algorithm #2: reverse time processing, function representation as cubes, cube extraction by means of a Podem-like implicit enumeration procedure.

The verification of complete hardware designs, which most often have much more than 60 state variables, can due to complexity problems not yet be verified by these techniques. As no single technique is clearly superior for all applications, further research for the appropriate combination of algorithms and exploitation of the circuit structure is required here.

3.4 Parameterized Hardware Modules. (4)

In contrast to control dominated circuits, data path dominated circuits are characterized by much more regularity in their implementations. They are often implemented in an iterative or recursive way as e.g. 32-bit ALU. VLSI module libraries for such data path elements are usually realized as parameterized module generators [2]. The hardware modules generated by such module generators are characterized by *potential complexity* and by *parameterizability*: The size of the combinatorial blocks as generated by such module generators can give rise to unmanageable complexities for checking their correctness by tautology checking methods as described in subsection 3.2. Several data path dominated design problems expose regularity and are defined in a parameterized way. Starting from a generic description several design instances can be generated. This makes them very suitable for the *formal proofs by induction*, for which general purpose theorem provers can be used very well.

Within CHARME, formal correctness proofs of parameterized module generators [31] as well as regular hardware structures [32,33] based on the Boyer-Moore [60] theorem prover, have been worked out successfully.

The formal proof by means of the Boyer-Moore theorem prover [60] has been successfully integrated with a module generation environment as used by the CATHEDRAL silicon compiler [31]. This has up to now allowed the discovery of more than 25 design and specification bugs that were previously uncovered by traditional simulations on the designs.

Other theorem provers and proof assistants have been investigated for hardware verification and compared: OTTER [34,35], HOL⁷ [59,36,37] and OBJ3 [38,62].

From the experience of the CHARME partners in the usage of general purpose theorem provers it has become clear that such tools have their specific strengths and weaknesses in comparison to more dedicated approaches such as those based on OBDD's and FSM verification as explained in subsections 3.2 and 3.3. General purpose provers have the advantage of a *uniform formalism*, the concept of *abstraction* and *proofs by induction*. The uniform formalism has the advantage that it allows reasoning in the formalism itself. But it has the disadvantage of being too general because they usually support the reasoning in some branch of mathematics instead of with the concepts and objects under design. To be useful in design this requires a policy for use of such provers. Even for theorem provers that have some automatic decision procedures, it has been experienced that a *lot of user interaction* and *system expertise* is required in the use of theorem provers. This means that in the application of theorem provers in hardware design, these tools should only be used where this investment can pay off. The aspect of *proofs by induction* is best exploited in parameterized hardware modules as well as in the correct definition of synthesis primitives. The aspect of abstraction can best be used at the higher level of design abstraction. Methodologies for design which rely on theorem provers should include the usage of theorem provers only at places where they do not require intervention of the day to day hardware designers.

3.5 Instruction Sets. (5)

As far as special purpose devices are concerned, between the SFG/Algorithmic level and the bRT level, control dominated subsystems require scheduling of the tasks to be executed in the hardware (data paths, I/O units, memories, etc.). This task is called scheduling [1,2].

On the other hand, with respect to microprocessors, the "instruction set" level is the highest level of abstraction and therefore an "instruction set" description can be considered as the microprocessor specification. The microprocessor implementation, that has to be checked versus this specification, can be described at various less abstract levels, from the "micro-sequence" level to the "data path" level [39,40]. Starting from the "instruction set" level, the proof consists in verifying, among two descriptions given at two adjacent levels, that the more detailed one is a correct implementation for the more abstract one. The lowest levels of description are written in VHDL, but for the highest ones, no appropriate single HDL exists. Therefore, in the CHARME project, a functional semantics [40] based on the \mathbf{P} -calculus [64] has been defined for each of these highest levels [39,40]; these definitions have been made in accordance with the formalisms and the principles of the tools that are used to achieve the proofs. Because

⁷ cooperation with the CHEOPS ESPRIT-BRA 3215

of the complexity of this semantics, the realization of an associated hardware description language is not desirable; rather, a user-friendly interactive editor can help the designer in describing his/her microprocessor.

Such a special purpose interactive tool (with windows and menus), called μ SPEED, has been developed for the specification and symbolic verification of (instruction sets of) microprocessors [41]. This tool constructs automatically the appropriate functional models of the microprocessors. This methodology has been used to successfully formulate the instruction sets of commercial microprocessors.

3.6 System Level Verification. (6)

In cooperation with research efforts in the CHEOPS project, a new system verification methodology called *SFG-Tracing* has been defined [42]. This verification methodology aims at the formal verification of designs across levels of abstraction. It is based on the observation that higher levels of abstraction are less detailed in their specifications (in terms of hardware and in terms of time instances). Therefore *SFG-Tracing* uses the higher level specification as a starting point and relies on the partitioning of this high level specification. In case of an SFG representation (but others are possible as well), the partitioning results in a number of boundary signal values, which are called *reference signals*. In the *SFG-Tracing methodology* it is required that the *mapping functions* in space and in time of these reference signals with respect to the lower level implementation are known. This methodology has resulted in the *full verification* of the transistor circuits as extracted from the layout with respect to the high level specification of a 32.000 transistor modem chip [58] as synthesized by CATHEDRAL-2 (see fig. 3).

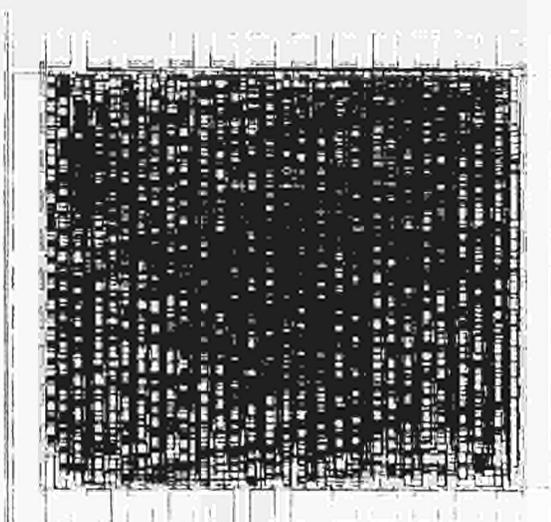


Figure 3: Chip layout of a 32000 transistor modem receiver pulse shaper and equalizer chip as synthesized by CATHEDRAL-2. This chip has 3 ALU's of 14 bits, register files, a multi-branch controller, micro sequencer, and testability circuitry. This chip has been fully verified w.r.t. high level specification using the *SFG-Tracing* methodology.

The complete interaction of sequential systems is captured in the *SFG-Tracing* verification methodology in contrast to only the individual combinatorial blocks at the SRT level as the methods explained in subsection 3.2. Due to the systematic partitioning of the specifications, *SFG-Tracing* does also not suffer from the complexity limits (around 60 state variables or 10^{20} states) as is the case in black box FSM verification

methods as explained in subsection 3.3. The chip in fig.3 contains 852 latches corresponding to 10^{256} states.

3.7 Asynchronous Subsystems. (7)

As explained under 2.1.1 almost all integrated circuits communicate with one another via asynchronous interfaces. The valid communication among such subsystems therefore needs verification. Whereas synchronous subsystems are more naturally described in *value-based* formalisms, asynchronous subsystems lend themselves more naturally to *event-based* formalisms. This is motivated by the asynchronous communication protocols.

In CHARME the *event-based* formalism CIRCAL [44] is being implemented for the formal verification of asynchronous subsystems [45] as well as its use for the integration in the synchronous circuits. CIRCAL is based on the concept of process algebras as it has been developed in the area of software engineering. Abstraction mechanisms for such an event based formalism have been worked out in CHARME [46]. Formal verification results are presented in [47].

3.8 Interface to VHDL as a Hardware Description Language. (8)

As the methodologies for formal verification are still under development, investigation, prototyping and comparison, currently all of the research efforts in CHARME as depicted in the *Formal Verification Map* of figure 2 are based on in-house hardware description languages and specific formalisms. This is motivated by the availability of previous research results and tools, the concentration on finding appropriate solutions of the design verification problem and by the available benchmarks.

However, to guarantee that what is being verified is indeed what has been designed, and gain acceptance in the community of hardware designers, formal verification should take as input the same user interface descriptions which are used for the other design tasks. VHDL [5] tends to become a "lingua franca" for hardware description in a significant part of the western world. Due to the considerable interest raised in the community by VHDL-based software, it is clear that dedicated efforts to concentrate on the formal verification of VHDL descriptions are required.

In CHARME, research work is being conducted to that goal. Unfortunately, VHDL is both a complex language, and a language for which no formal semantic definition is provided. Our pragmatic attitude has consisted in selecting a significant VHDL subset for which we have defined the semantics in terms of a formally manipulable model, and writing a translator from VHDL to the input format of provers for this model (this work has taken advantage of the translation principles presented in [61]). This work has led to the implementation of a prototype of Formal Proof Environment for VHDL, which is called PREVAIL [48,50], and which checks the functional equivalence between two "architecture" bodies, describing alternative implementation hypotheses, or different description levels, for the same design "entity", where one is considered as the specification, and the other one is the implementation. In order to keep the proof problem within manageable size, we take the well known "divide and conquer" strategy, and have identified modelling levels and circuit types, together with a preferred description style in VHDL, for which particular proof tools appear to be efficient. More specifically, this is done in the current status of our PREVAIL environment by defining:

- A syntactic subset of VHDL for Zero-delay combinational circuits; for this subset, a restricted and simplified semantics is associated to the VHDL architecture.
- A syntactic subset and description styles for the synchronous sequential circuits, and their associated functional semantics.

Nevertheless, the current status of this prototype does not allow the processing of the VHDL timing constructs such as "stable" (except for the master clock), "last-event", and thus avoids reasoning on asynchronous primitives. To that goal, recent work concentrates on the formal definition of the semantics of the VHDL timing constructs, and on the verification of associated properties (given in an informal way in the LRM) by means of the Boyer-Moore Theorem Prover [49].

The preliminary conclusions of all that work are that some concepts (such as memorization, FSM's) must be clearly defined in VHDL in order to make formal reasoning more manageable. This has already given rise to a number of recommendations for the 1992 ballot on the new VHDL standard.

4 Design for Verifiability.

Given the fact that formal verification still needed to go a long way to become useful in practice, the CHARME partners decided in 1988 to define guidelines under which formal verification of hardware could become practical. In analogy with *Design for Testability* (DfT) [6], this has been called *Design for Verifiability* by the partners.

DfV such as also DfT has to take into account the possibilities of the technology enabling formal verification. With respect to DfV there are basically two perspectives to look at the problem. One aspect is to define a number of rules or constraints to which designs have to adhere in order to become verifiable [51,53]. An other aspect is to concentrate on the verification *methodology* to be introduced in the design process [52,54].

5 Directions for Future Research.

The Formal Verification Map represents an overall classification of aspects and levels of abstraction on which formal verification tools are being worked out. As indicated in the previous section large progress has been made in nearly all of the classes in the last few years. However still a lot of efforts are required to extend the possibilities of formal verification as well as to introduce formal verification methods in the every day industrial hardware design process.

In the future, further consideration is necessary in order to automate the formal verification process. This requires further elaboration of the basic technology of Boolean comparison as outlined in section 3.2. Multipliers as often occur in VLSI hardware are still a problem [24] when represented as OBDD's. Initial approaches in this direction have been investigated already [55,63]. Although algorithms exist for generating the orders of variables in OBDD's [21,22] further research is required to exploit the regularity and structure of high level specifications in generating such orders.

In the automatic verification of combinatorial functions, further algorithms are required for exploiting the vector aspects in the expressions. Initial techniques in this direction have been worked out in CHARME [57].

Because the generated mask layouts are one of the formal hardware representations that are the nearest as a model to the actual circuits being realized, the formal verification starting from the layout extracted transistor circuits is extremely important. Therefore,

on the switch level of abstraction, future efforts have to concentrate on the efficient symbolic analysis and evaluation of transistor circuits with over 1.000.000 transistors.

As stated previously, the validation of the usage of the switch-level model deserves further attention in future research.

The validation of external properties of specifications, as also used in synthesis should be worked out further. Current approaches in model checking only concentrate on the verification of systems with around 60 state variables [29,30], which is still far from actual circuits. To cope with realistic complexities, more of the structure at hand in hardware specifications has to be exploited. In the industrial practice synthesis from high level specifications is becoming more and more accepted. This allows designers to make fast tradeoffs between different implementation alternatives. It should however be made sure that these specifications still meet the original requirements and specifications.

As general purpose theorem provers require a large amount of human expertise and interaction their usage should be restricted to those places where their use can really enlarge the quality of designs at hand as well as where "the regular designer" can be hidden from their intervention. This can be mainly exploited in the formal design of parameterized library entries as well as alternative synthesis primitives in libraries [56]. The aspect of abstraction is being exploited in transformational design systems.

Formal languages as used in theorem provers [59,60] can be used for the consistent formal definition of operator primitives that are used in CAD environments for purposes of *simulation*, *verification* and *synthesis*.

System level verification methodologies such as *SFG-Tracing* are as a methodology not restricted to the levels of abstraction to which they are currently used in the CHARME and CHEOPS projects. Indeed alternative representations such as VHDL and even 'C-code' models require future investigation.

In the implementation of complex systems, trade-offs have to be made on which parts to be made in hard- and which parts to be made in software. Future research has to concentrate on making this migration possible in an *efficient* and *correct* way.

VHDL is the HDL for the years to come. Even though reservations could be made with respect to certain aspects of the language, formal verification tools will have to adopt languages such as VHDL in order to be able to introduce formal verification methodologies in the actual hardware design trajectory.

In comparison to the synchronous subsystems, the aspect of the correct synthesis and verification of asynchronous subsystems needs further elaboration, because the correctness of large systems interconnected via asynchronous interfaces are critically dependent on the correctness of these interfaces for the correctness of the global systems.

Several of these future aspects will be the target of the CHARME-II project.

6. Conclusions.

In this paper an overview has been given of the broad field of formal verification in relation to design *aspects* and *abstraction levels*. This is necessary in order to understand and to be able to compare the specific approaches to address specific problem areas. A large progress has been made in the whole formal verification field. Having a mixed background cooperation in CHARME of electrical engineers, computer scientists and mathematicians has enabled to put together the appropriate ingredients to migrate formal verification from theory in industrial practice. This is already illustrated by the practical results of the full verification of actual VLSI chips from the transistor

level as extracted from the layout with respect to the high level specifications. This is *the largest full verification of a complete integrated circuit done thus far*. Previous approaches in the formal verification of e.g. microprocessors have all skipped important levels in the design abstractions. Also the adoption of designer oriented description languages such as VHDL are a key to introducing formal verification methods in the industrial practice.

In a fast growing field as formal design verification, the CHARME partners are stimulating the communication of researchers in the field by organizing state-of-the-art public workshops. These have already been organized in Darmstadt, Glasgow, Grenoble, Leuven and Turin. These meetings assist in the further comparison and synergy of promising approaches in the area of formal verification.

7. Acknowledgements.

The authors hereby thank the EEC for sponsoring this focused research project, which has enabled further progress and superior results, that would not have been possible in individual efforts only. They also would like to acknowledge the direct or indirect contributions of the following researchers to the implementation and elaborations of the ideas and results being developed inside the CHARME project: M. Allemand, C. Angelo, A. Bailey, C. Bayol, P. Camurati, P. Cockshott, H. Collavizza, D. Deharbe, P. De Vijt, M. Genoe, M. Gilli, S. Hoereth, B. Huber, C. Le Faou, W. Lempens, W. Mao, T. Margaria, G. McCaskill, G. Milne, L. Pierre, W. Ploegaerts, F. Proesmans, A. Salem, H. Samsom, M. Sonza Reorda, U. Schellin, J. Vandenberg, D. Verkest, E. Verlind. The authors thank the reviewers for their constructive comments.

References

- [1] M.C. McFarland, A.C. Parker, R. Camposano, "The High-Level Synthesis of Digital Systems", *Proceedings of the IEEE*, Vol. 78, No. 2, February 1990, pp.301-318.
- [2] H. De Man, J. Rabaey, P. Six, L. Claesen, "Cathedral-II: A silicon compiler for digital signal processing", *IEEE Design & Test of Computers*, December 1986, Vol. 3, No. 6, pp.73-85.
- [3] J.N. Reed, A.W. Roscoe, "Technology Study: Formal Methods for the Development of Computer Systems", *EuroTechnology*, Issue No.9, April 1991, Blackwell Professional Information Services, ISSN 0959-7735, pp. 12-14.
- [4] P. Camurati, P. Prinetto, "Formal verification of hardware correctness: introduction and survey of current research", *IEEE Computer*, July 1989, pp. 8-19.
- [5] -, "IEEE Standard VHDL Language Reference Manual", IEEE Standard 1076-1987.
- [6] H. Fujiwara, "Logic Testing and Design for Testability", *Computer System Series*, The MIT Press, ISBN 0-262-06096-5, 1985.
- [7] R.E. Bryant, "A Switch-Level Model and Simulator for MOS Digital Systems", *IEEE Transactions on Computers*, Vol. C-33, No.2, February 1984, pp. 160-177.
- [8] R.E. Bryant, "Algorithmic aspects of symbolic switch network analysis", *IEEE Transactions on Computer-Aided Design*, Vol. CAD-6, No. 4, July 1987, pp. 618-633.
- [9] R.E. Bryant, "Boolean Analysis of MOS Circuits", *IEEE Transactions on Computer-Aided Design*, Vol. CAD-6, No. 4, July 1987, pp. 634-649.
- [10] S. Hoereth, "Improving the performance of a BDD-based tautology checker", *Proc. Advanced Research Workshop on Correct Hardware Design Methodologies*, ed. P. Prinetto, P. Camurati, Turin, June 12-14, 1991.

- [11] P. Herrebut, "BOTRYS: A program for the symbolic analysis of MOS circuits at the switch level", Thesis IMEC - Katholieke Universiteit Leuven Belgium, July 1988.
- [12] W. Lempens, "Symbolic analysis of digital MOS circuits at the switch level", Thesis IMEC Katholieke Universiteit Leuven Belgium, July 1989.
- [13] R.E. Bryant, D. Beatty, K. Brace, K. Cho, T. Sheffer, "COSMOS: A Compiled Simulator for MOS Circuits", 24th Design Automation Conference, pp. 9-16, 1987.
- [14] H. Eveking, "Behavioral consistency between register-transfer and switch-level descriptions", *Proc. IFIP TC-10 Working Conf. on Design Methodologies for VLSI and Computer Architecture*, pages 183-202, North-Holland, 1988.
- [15] H. Eveking, "Behavioral verification of synchronous systems" in G. Milne and P.A. Subrahmanyam editors, *Formal Aspects of VLSI Design*, pages 137-152, North-Holland, 1985.
- [16] H. Eveking, "Axiomatizing hardware description languages", *International Journal of VLSI Design*, Vol. 2, Nr. 3, 1990.
- [17] D. Verkest, L. Claesen, "Special Benchmark Session on Tautology Checking", *Formal VLSI Correctness Verification*, ISBN 0 444 88688 5, North-Holland Elsevier Science Publishers, 1990, p.81-82.
- [18] C. Bayol, J.-L. Paillet, "Using TACHE for proving circuits", *Formal VLSI Correctness Verification*, ed. L.Claesen, ISBN 0 444 88688 5, North-Holland Elsevier Science Publishers, 1990, p.83-87.
- [19] R.E. Bryant, "Graph Based Algorithms for Boolean Function Manipulation", *IEEE Transactions on Computers*, Vol. C-35 No. 8, August 1986, pp. 667-691.
- [20] J.C. Madre, J.P. Billon, "Proving Circuit Correctness using Formal Comparison Between Expected and Extracted Behavior", *Proc. of the 25th Design Automation Conference*, 1988.
- [21] M. Fujita, H. Fujisawa, N. Kawato, "Evaluation and Improvements of Boolean Comparison Method Based on Binary Decision Diagrams", *Proc. IEEE ICCAD-88 Conference*, 1988, pp. 2-5.
- [22] S. Malik, A.R. Wang, R.K. Brayton, A. Sangiovanni-Vincentelli, "Logic Verification using Binary Decision Diagrams in a Logic Synthesis Environment", *Proc. IEEE ICCAD-88 Conference*, 1988, pp. 6-9.
- [23] F. Anceau, in *panel session: Formal Hardware Verification: Myth or Reality*, EDAC-91, Amsterdam, February 25-28, 1991.
- [24] R.E. Bryant, "On the Complexity of VLSI Implementations and Graph Representations of Boolean Functions with Application to Integer Multiplication", *report Carnegie Mellon University*, September 27, 1988.
- [25] P. Camurati, M. Gilli, P. Prinetto, M. Sonza Reodra, "Model Checking and Graph Theory in sequential ATPG", *Workshop on Computer-Aided Verification*, June 1990, Rutgers, NJ (USA).
- [26] P. Camurati, e. a., "The Product Machine and Implicit Enumeration to prove FSMs Correct", *Proc. Advanced Research Workshop on Correct Hardware Design Methodologies*, June 12-14, 1991, Turin, Italy, pp. 305-319.
- [27] O. Coudert, C. Berthet, J.-C. Madre, "Verification of Sequential Machines Using Functional Vectors", *Formal VLSI Correctness Verification*, ISBN 0 444 88688 5, North-Holland Elsevier Science Publishers, 1990, p.179-196.
- [28] H. Eveking, "Experience in Designing Formally Verifiable HDL's", *Proc. Computer Hardware Description Languages and their Applications CHDL-91*, ed. D. Borrione, R. Waxman, Marseille, April 22-24, 1991, pp. 301.
- [29] S. Bose, A. Fisher, "Automatic Verification of Synchronous Circuits using Symbolic Logic Simulation and Temporal Logic", *Formal VLSI Correctness Verification*, ISBN 0 444 88688 5, North-Holland Elsevier Science Publishers, 1990, p.151-158.

- [30] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, "Sequential Circuit Verification Using Symbolic Model Checking", *Proc. Design Automation Conference, DAC-90*, June 24-28, 1990, pp. 46-51.
- [31] D. Verkest, L. Claesen, H. De Man, "Correctness proofs of parameterized hardware modules in the Cathedral-II synthesis environment", *Proc. European Design Automation Conference EDAC-90*, Glasgow 12-15 March 1990.
- [32] L. Pierre, "The Formal Proof of Sequential Circuits described in CASCADE using the Boyer-Moore Theorem Prover", and "The Formal Proof of the "Min-max" sequential benchmark described in CASCADE using the Boyer-Moore Theorem Prover", *Formal VLSI Correctness Verification*, ISBN 0 444 88688 5, North-Holland Elsevier Science Publishers, 1990, p.309-348.
- [33] L. Pierre, "One Aspect of Mechanizing Formal Proof of Hardware: The Generalization of Partial Specifications", *Proc. 1991 International Workshop on Formal Methods in VLSI Design*, ed. P.A. Subrahmanyam, ACM/SIGDA, Miami Florida, January 9-11, 1991.
- [34] P. Camurati, T. Margaria, P. Prinetto "Use of the OTTER theorem prover for the formal verification of hardware", *Euromicro'90*, August 1990, Amsterdam, (The Netherlands).
- [35] P. Camurati, T. Margaria, P. Prinetto, "Resolution-based correctness proofs of synchronous circuits", *Proc. European Design Automation Conference EDAC-91*, IEEE Computer Science Press, Amsterdam, 25-28 February 1991, pp. 11-15.
- [36] C. Angelo, L. Claesen, H. De Man, "A Methodology for Proving Correctness of Parameterized Hardware Modules in HOL", *proc. Tenth International Symposium on Computer Hardware Description Languages and their Applications, CHDL-91*, Marseille, April 22-24.
- [37] C. Angelo, D. Verkest, L. Claesen, H. De Man, "On the comparison of HOL and Boyer-Moore for formal hardware verification", *proc. Advanced Workshop on Correct Hardware Design Methodologies*, ed. P. Camurati, P. Prinetto, Turin, June 12-14.
- [38] H. Collavizza, L. Pierre, "Basic Verification Techniques - Evaluation of General Provers", *ESPRIT CHARME Report: UP-2.A.2.-OI*, 30 June 1990.
- [39] H. Collavizza, "Functional Semantics of Microprocessors at the Micro-Program level and Correspondance with the Machine Instruction Level", *proc. of the EDAC Conf.*, Scotland, 12-15 March 1990, pp. 52-56.
- [40] J.L. Paillet, "Functional Semantics of Microprocessors at the Machine Instruction Level", *Proc. 9th IFIP Int. Conf. CHDL*, North-Holland, Washington D.C., June 1989.
- [41] D. Borrione, H. Collavizza, C. Le Faou, "μSPEED: a Framework for Specifying and Verifying Microprocessors", *Proc. 1991 International Workshop on Formal Methods in VLSI Design*, ACM/SIGDA, Miami, January 9-11, 1991.
- [42] L. Claesen, F. Proesmans, E. Verlind, H. De Man, "SFG-Tracing: a Methodology for the Automatic Verification of MOS Transistor Level Implementations from High Level Behavioral Specifications", *Proceedings A CM-SIGDA International Workshop on Formal Methods in VLSI Design*, ed. P.A. Subrahmanyam, January 9-11, 1991.
- [43] M. Genoe, L. Claesen, E. Proesmans, E. Verlind, H. De Man, "Illustration of the SFG-Tracing Multi-Level Behavioral Verification Methodology, by the Correctness Proof of a High to Low Level Synthesis Application in CATHEDRAL-11", *Proc. IEEE ICCD-91*, Conference, Cambridge MA, October 14-16, 1991.
- [44] G.J. Milne, "Circal and the representation of communication, concurrency and time", *ACM Trans. on Programming Languages and Systems*, 7(2), 1985.
- [45] A. Bailey, G. Milne, "Using CIRCAL to Analyse Sutherland's Asynchronous Micropipeline Design Style", *Dept. of Computer Science Research Report*, HDV-13-91, University of Strathclyde, UK, May 1991.
- [46] A. Bailey, "Abstraction Mechanisms for Hardware Verification: Formalisation in a Process Algebra", *Proc. CHDL-91*, editors: D. Borrione, R. Waxman, Marseille France, April 22-24, 1991.

- [47] A. Bailey, G.A. McCaskill, J. McIntosh, G.J. Milne, "The description and automated verification of digital circuits in Circal", *Proc. Advanced Research Workshop on Correct Hardware Design Methodologies*, ed. P. Prinetto, P. Camurati, Turin, June 12-14, 1991.
- [48] D. Borrierc, A. Salem, "Design For Verifiability - Automatic Formal Verification of VHDL descriptions", *ESPRIT CHARME report UP-I.C-01*, 30 June 1990.
- [49] A. Salem. D. Borrierc, "Formal Semantics of VHDL Timing Constructs", *Proc. EURO-VHDL'91*, Stockholm, 8-11 September 1991.
- [50] D. Borrierc, L. Pierre, A. Salem, "PREVAIL: A Proof Environment for VHDL Descriptions", *Proc. Advanced Research Workshop on Correct Hardware Design Methodologies*, ed. P. Prinetto, P. Camurati, Turin, June 12-14, 1991.
- [51] G. Milne, "Design for Verifiability", In *Proc. Workshop on Hardware Specification, Verification and Synthesis: Mathematical Aspects*, Cornell Univ. 1989.
- [52] H. Evekings, "Preliminary Concepts of Design for Verifiability", *ESPRIT CHARME Report THDI.C-01*, July 20, 1990.
- [53] P. Camurati, P. Prinetto, "Design for Verifiability and Design for Testability: limiting designers' freedom to achieve what?", *Proc. Advanced Research Workshop on Correct Hardware Design Methodologies*, ed. P. Prinetto, P. Camurati, Turin, June 12-14, 1991.
- [54] L. Claesen, M. Genoe, E. Verlind, F. Proesmans, H. De Man, "SFG-Tracing: a methodology of Design for Verifiability", *Proc. Advanced Research Workshop on Correct Hardware Design Methodologies*, ed. P. Prinetto, P. Camurati, Turin, June 12-14, 1991.
- [55] H. Simonis, "Formal Verification of Multipliers", *Formal VLSI Correctness Verification*, ISBN 0 444 88688 5, North-Holland Elsevier Science Publishers, 1990, p.267-286.
- [56] D. Verkest, J. Vandenbergh, L. Claesen, H. De Man, "Formal Design and Verification Strategy of Parameterized Hardware Modules", *Proc. Advanced Research Workshop on Correct Hardware Design Methodologies*, ed. P. Prinetto, P. Camurati, Turin, June 12-14, 1991.
- [57] A. Bratch, H. Evekings, H.-J. Faerber, J. Pinder, U. Schellin, "LOVERT - A Logic Verifier of Register Transfer Level Descriptions", *Formal VLSI Correctness Verification*, ISBN 0 444 88688 5, North-Holland Elsevier Science Publishers, 1990, p.247-256.
- [58] J. Vanhoof, I. Bolsens, S. De Troch, E. Blokken, H. De Man, "Evaluation of high-level design decisions using the Cathedral-11 silicon compiler to prototype a DSP ASIC", *Proceedings, IFIP Workshop on High Level and Logic Synthesis*, ed. G. Saucier, Paris, 30 May-1 June 1990.
- [59] M.J.C. Gordon, "HOL: A Proof Generating System for Higher-Order Logic", in *"VLSI Specification, Verification and Synthesis"* ' editors: G. Birtwistle and P.A. Subrahmanyam, Kluwer 1987.
- [60] R.S. Boyer, J.S. Moore, "A Computational Logic Handbook", *Academic Press*, Boston, 1988.
- [61] L. Pierre, "From a HDL Description to Formal Proof Systems : Principles and Mechanization", *Proc. 10th IFIP Int. Conf. CHDL'91*, Ed. D. Borrierc & R. Waxman, Marseillc, 22-24 April 1991.
- [62] J. Goguen, "OBJ as a Theorem Prover with Applications to Hardware Verification", *Technical report SRI-CSL-88-4R2*, Computer Science Laboratory, SRI International, Menlo Park, August 1988.
- [63] J. Burch, "Using BDDs to Verify Multipliers", *Proc. 28th ACM/IEEE Design Automation Conference*, San Francisco (CA), 17-21 June 1991.
- [64] J.L. Paillet, "A Functional Model for Descriptions and Specifications of Digital Devices", *Proc. IFIP Int. Working Conf. "From HDL Descriptions to Guaranteed Correct Circuit Designs"*, Grenoble, September 1986.

[65] D. Borrione, D. Deharbe, H. Eweking, St. Horeth, "Application of a BDD-package to the verification of HDL-descriptions", *Proc. Advanced Research Workshop on Correct Hardware Design Methodology*, Turin, 12-14 June 1991.

INDEX OF AUTHORS

Author	Page	Author	Page
Acid S.	363	Clausen T.C.	646
Acket G.A.	135	Cluet S.	758
Addison C.	352	Coupric M.	191
Alnsworth W.	222	Coutaz J.	707
Ambrosius H.P.M.	135	Crane S.	207
Angéniol B.	393	Crouzet Y.	234, 791
Arlat J.	234, 791	Cunningham J.	822
Arrighetti S.	322		
Atkinson M.	731	Davies D.R.H.	675
		Day R.A.	512
Babini G.	222	de Campos L.M.	363
Bann S.	264	de Lange M.	18
Bardyn J.-J.	512	de Maquillé Y.	101
Bargain R.	101	de Mey V.	534
Basaglia, G.	34	de Moor D.	661
Beal D.	18	Delobel C.	758
Beaumont S.	80	Di Benedetto M.-G.	378
Becker T.	455	Duce R.A.	512
Belloti V.	720	Dulay N.	207
Ben Amara H.	303	Dümcke R.	101
Benyattou T.	773, 780		
Bergsten B.	191	Eberhard G.	529
Bernardo L.M.	601	Efthimiadis A.	593
Berrino C.	692	Engelhardt K.	439
Bishop N.	352	Eveking H.	857
Borrionel D.	857	Exposito J.	152
Brandenburger J.	101		
Brenner K.H.	264	Faillot J.-L.	563
Brissot L.	439	Fanshawe D.G.J.	506
Brocklehurst S.	806	Fernandez-Gonzalez F.	578
Brunner H.	439	Ferrari F.	322
Buchkremer H.St.	264	Finger M.	288
Bulthuis W.	553	Fissore L.	222
Burrill L.	234	Fogelman F.	393
Bush M.	278	Frangoulis E.	222
Buxton B.F.	322	Friedel P.	393
		Frijlink P.M.	135
Cadore R.	780		
Cahill V.	427	Gabbay D.	822
Calleja Pardo E.	135	Gadga C.	439
Castelow D.A.	322	Gale M.T.	439
Chandler N.	101	Gallop J.R.	512
Chantraine P.	101	Garcia Gomez R.	222
Chappel H.	303	Gendry M.	773
Chapuis C.	615	Gil E.	780
Cheung S.C.	207	Giry Ph.	393
Chilo J.	101	Gonzalez A.	363
Cilia G.	439	Goodman D.	675
Claes A.	152	Graf S.	234
Claesen L.	857	Grenier Y.	222
Clark S.A.	773	Grimsdale R.	18

Author	Page	Author	Page
Guida M.	34	Lohman G.	264
Guillot G.	773, 780	MacInnes A.	234
		Maclean A.	720
Halewood S.J.	601	Magée J.	207
Harris C.G.	322	Magrassi M.	322
Hauck R.	601	Makram S.	393
Hernandez-Gil Gomez J.F.	135	Mantzari P.	593
Heuschötter P.	479	Manuello D.	692
Hey T.	352	Marcadé E.	393
Heyer G.	470	Marchal J.M.	135
Hiller S.M.	378	Martins E.	234
Hockney R.	352	Masciangelo S.	322
Höge M.	470	Mazzocchi G.	601
Hollan L.	135	McBrien P.	288
Hollinger G.	773	McLauchlan P.	322
Hoppe J.	409	Medina M.	421
Horn C.	427	Mellor P.	806
Huignar J.P.	264	Meschenmoser R.	615
		Metge S.	806
Jennings N.R.	253	Metzler P.	439
Jensen A.S.	264	Mihailovic M.	780
Joffre P.	264	Mistral L.	512
Juergensen H.	135	Molina R.	363
		Mcran T.	720
Kanoun K.	234, 806	Moreau JP.	174
Keil K.	479	Moreno A.	421
Kelly G.	152	Moron P.	322
Kese R.	470		
Klein A.	336	Nachtsheim R.	479
Knight G.	675	Neef G.	152
Kordey N.	479	Nierstrasz O.	534
Korte W.B.	479	Nigay L.	707
Koschnik W.	101		
Kramer J.	207	O'Mathuna C.	101
Kritter S.	121	Ohlbach H.J.	822
Kroszynski U.I.	646	Owens R.	288
Kugler M.	470		
		Paillet J.L.	857
Lang G.K.	439	Papafotiou K.	593
Laprie J.-Cl.	791, 806	Perez de la Blance N.	363
Larintzakis M.	593	Peroche B.	303
Laver J.	378	Philbrow, P.	731
Lea R.	427	Piffault N.	780
Leclerc P.	18	Pimont J.-M.	393
Lécluse C.	731	Powell D.	234
Lefèvre J.-P.	378	Powell K.	601
Lehne M.G.	578	Prinetto P.	857
Leone V.	439	Pruijmboom A.	91
Lescure M.	615		
Leuridan J.	628	Rahaga T.	121
Leymarie J.	780	Rajbenbach H.	264
Littlewood B.	806	Ramacher U.	393
Lockwood Ph.	222	Rasmussen E.	264

Author	Page	Author	Page
Raynor J.M.	439	Treleaven P.	393
Recce M.	393	Trostmann E.	646
Refregier Ph.	264	Tsichritzis D.	534
Richard Ph.	731	Twidle K.	207
Richier J.-L.	234		
Robert Ph.	3	v. Kleist-Retzow B.	470
Robinson S.	479	Vaillant R.	322
Romaguera J.	675	Valduriez P.	191
Rooney E.	378	van der Auweraer H.	628
Roviras D.	615	Vandeloo P.	112
Russell M.	278	Vandeurzen U.	628
Rygol M.	322	Vasson A.	780
		Vasson A.M.	780
Sandini G.	322	Verhelst B.	112
Schützeneder H.	529	Viktorovitch P.	773
Seitz P.	439	Voiron J.	234
Simon C.	393	von der Nuell S.	322
Sloman M.	207	Vopel R.	578
Sobiesierski Z.	773	Wahl M.	112
Sørensen T.	646	Wang H.	322
Sousa P.	427	Waucquez Ch.	135
Stadelmann M.	534	Wieczorek R.	18
Starovic G.	427	Williams R.H.	773
Strasser W.	18	Wilson M.D.	303
Strauch G.	135	Winkelmann G.	470
Sutcliffe D.C.	512	Wolton I.	352
		Wortmann J.C.	842
Tabata A.	773, 780		
Tanner A.	806	Young R.	720
Theeten J.B.	393		
Theiler T.	165	Zachariassen K.	152

INDEX OF PROJECT NUMBERS

Project Number	Page	Project Number	Page
0710	675	2479	121
1561	563	2484	18
2001	409	2486	628
2010	578	2502	322
2016	91	2614	646
2025	191	2623	661
2071	421, 427	2701	336
2075	101	2702	352
2080	207	2704	529
2090	593	2705	534
2101	222	3066	707, 720
2103	439	3070	731, 758
2127	601	3086	773, 780
2146	455	3092	791, 806
2151	3	3125	822
2198	615	3133	80
2252	234	3143	842
2256	253	3216	857
2288	264	5003	135
2315	470	5033	152
2318	112	5041	165
2382	479	5075	174
2384	278	5170	363
2431	506	5192	378
2434	34	5293	393
2463	512	5386	54
2470	288	5656	553
2474	303		

INDEX OF ACRONYMS

Acronym	Page	Acronym	Page
AMODEUS	707, 720	LDS	773, 780
APACHIP	101		
ARCHON	253	MAGIC	661
ARGOS	512	MASCOT	439
ARS	222	MEDLAR	822
ASCODY	563	METKIT	278
		MMI2	303
BASE-TIP	91		
		NANSDEV	80
CHARME	857	NAOPIA	264
COMANDOS-2	421, 427	NEUTRABAS	578
COSINE	675	NIRO	646
CRYPTOCARD	529		
		OMI-MAP	54
DAMS-2	455	OSMOSE-1	553
DELTA-4	234		
DYNAMO	628	PDCS	791, 806
		PLANET	135
EDS	191	PLASIC	152
ELO	479	PROMIMPS	165
EPIC	593	PUMA	336
EVEREST	112		
		REX	207
FCPN	615		
FIDE	731, 758	SCOPE	3
FOF	842	SPELL	378
		SPIRIT-1	18
GALATEA	393	SPRITE	409
GENESIS	352	STATLOG	363
HIDCIM	601	TEMPORA	288
HOME	506	TWB	470
IDPS-TIP	174	VOILA	322
ITHACA-2	534		

European Communities – Commission
EUR 13853 – Esprit '91 – Conference proceedings

Luxembourg: Office for Official Publications of the European
Communities

1991 – XI, 881 pp., num. tab., fig. – 16.2 × 22.9 cm

ISBN 92-826-2905-8

Catalogue number: CD-NA-13853-EN-C

Price (excluding VAT) in Luxembourg: ECU 65

Venta y suscripciones • Salg og abonnement • Verkauf und Abonnement • Πωλήσεις και συνδρομές
Sales and subscriptions • Vente et abonnements • Vendita e abbonamenti
Verkoop en abonnementen • Venda e assinaturas

BELGIQUE / BELGIË

Moniteur belge / Belgisch Staatsblad
Rue de Louvain 42 / Louvènweg 42
1000 Bruxelles / 1000 Brussel
Tél (02) 512 00 26
Fax 511 01 84
CCP / Postrekening 000-2005502-27

Autres distributeurs
Overige verkooppunten

Librairie européenne/

Europese Boekhandel

Avenue Albert Jonnart 50 /

Albert Jonnartlaan 50

1200 Bruxelles / 1200 Brussel

Tél (02) 734 02 81

Fax 735 06 60

Jean De Lannoy

Avenue du Roi 202 / Koningslaan 202

1060 Bruxelles / 1060 Brussel

Tél (02) 538 51 69

Télex 63220 UNIBOOK B

Fax (02) 538 08 41

CREDOC

Rue de la Montagne 34 Bergstraat 34

Bte 11 Bus 11

1000 Bruxelles / 1000 Brussel

FRANCE

Journal officiel
Service des publications
des Communautés européennes
26, rue Desaix
75727 Paris Cedex 15
Tél (1) 40 58 75 00
Fax (1) 40 58 75 74

IRELAND

Government Publications

Sales Office

Sun Alliance House

Molesworth Street

Dublin 2

Tel. (1) 71 03 09

or by post

Government Stationery Office

EEC Section

6th floor

Bishop Street

Dublin 8

Tel (1) 78 16 66

Fax (1) 78 06 45

UNITED KINGDOM

HMSO Books (PC 16)
HMSO Publications Centre
51 Nine Elms Lane
London SW8 5DR
Tel. (071) 873 2000
Fax GP3 873 8463
Télex 29 71 138

ÖSTERREICH

Mans'sche Verlags-

und Universitätsbuchhandlung

Kohlmarkt 16

1014 Wien

Tel. (0222) 531 61-0

Télex 11 25 00 BOX A

Fax (0222) 531 61-61

YUGOSLAVIA

Privredni Vjesnik
Bulevar Lenjina 171/XIV
11070 Beograd
Tel. (11) 123 23 40

CYPRUS

Cyprus Chamber of Commerce and

Industry

Chamber Building

36 Girvas Digenis Ave

3 Deligorgis Street

PO Box 1455

Nicosia

Tel. (2) 449500/482312

Fax (2) 458630

DANMARK

J. H. Schultz Information A/S

EF-Publikationer

Offitilavvej 18

2500 Valby

Tlf. 36 44 22 66

Fax 36 44 01 41

Girokonto 6 00 06 86

NOTICE TO THE READER

All scientific and technical reports published by the Commission of the European Communities are announced in the monthly periodical '**euro abstracts**'. For subscription (1 year: ECU 92) please write to the address below.

Price (excluding VAT) in Luxembourg: ECU 65



OFFICE FOR OFFICIAL PUBLICATIONS
OF THE EUROPEAN COMMUNITIES

L-2985 Luxembourg

ISBN 92-826-2905-8



9 789282 629055