Information Infrastructure Systems for Manufacturing

Visit the IT & Applied Computing Resource Centre at:
www.IT-CH.com
IFIP – The International Federation for Information Processing

IFIP was founded in 1960 under the auspices of UNESCO, following the First World Computer Congress held in Paris the previous year. An umbrella organization for societies working in information processing, IFIP’s aim is two-fold: to support information processing within its member countries and to encourage technology transfer to developing nations. As its mission statement clearly states,

IFIP’s mission is to be the leading, truly international, apolitical organization which encourages and assists in the development, exploitation and application of information technology for the benefit of all people.

IFIP is a non-profitmaking organization, run almost solely by 2500 volunteers. It operates through a number of technical committees, which organize events and publications. IFIP’s events range from an international congress to local seminars, but the most important are:

- the IFIP World Computer Congress, held every second year;
- open conferences;
- working conferences.

The flagship event is the IFIP World Computer Congress, at which both invited and contributed papers are presented. Contributed papers are rigorously refereed and the rejection rate is high.

As with the Congress, participation in the open conferences is open to all and papers may be invited or submitted. Again, submitted papers are stringently refereed.

The working conferences are structured differently. They are usually run by a working group and attendance is small and by invitation only. Their purpose is to create an atmosphere conducive to innovation and development. Refereeing is less rigorous and papers are subjected to extensive group discussion.

Publications arising from IFIP events vary. The papers presented at the IFIP World Computer Congress and at open conferences are published as conference proceedings, while the results of the working conferences are often published as collections of selected and edited papers.

Any national society whose primary activity is in information may apply to become a full member of IFIP, although full membership is restricted to one society per country. Full members are entitled to vote at the annual General Assembly. National societies preferring a less committed involvement may apply for associate or corresponding membership. Associate members enjoy the same benefits as full members, but without voting rights. Corresponding members are not represented in IFIP bodies. Affiliated membership is open to non-national societies, and individual and honorary membership schemes are also offered.
CONTENTS

Preface ix
Introduction: Towards an information infrastructure for manufacturing industry xi

PART ONE   Keynote and Invited Papers

1 Vehicle CALS – a big challenge to virtual development
   T. Mase 3
2 Changes in the industrial environment and CALS
   L. Zuijdeest 11
3 Perspectives on manufacturing engineering software integration
   C.R. McLean 20
4 Trends in manufacturing control
   P.J.M. van Bommel 32

PART TWO   Modelling of Products, Processes and Systems

5 Integration of product life cycle views on the basis of a shared conceptual information model
   R. Anderl and A. Wasmer 47
6 CIMOSA process model for enterprise modelling
   K. Kosanke, F.B. Vernadat and M. Zelm 59
7 A formal requirements engineering framework for CIM infrastructures reengineering
   M. Petit and E. Dubois 69
8 A supporting enterprise infrastructure using STEP technologies
   S.P. Reid and S.K. Banerjee 81
9 On reference modelling of integrated manufacturing systems using OOA
   G. Vosniakos and M. Williams 91
10 Metamodels for information system modeling in production environments
    G.S. Mannarino, G.P. Henning and H.P. Leone 103
11 Comparison of enterprise modelling methodologies
    K. Kosanke 115
12 Bottom-up modelling with CIMOSA
    W. Reithofer 128
13 Modelling and reorganizing of process chains using CIMOSA
    B. Janusz 140
PART THREE  Integration Frameworks and Architectures

14 The System Integration Architecture: a framework for extended enterprise information systems  
   J.J. Mills and M. Brand  155

15 A model based approach to enterprise wide information support  
   P.E. Clements, I.A. Coutts and J.M. Edwards  169

16 The Glue Logic: an integrated programming/execution environment for distributed manufacturing work-cell control system  
   M. Takata and E. Arai  181

17 A framework for connecting work and information infrastructure  
   J. Goossenaerts  193

18 An extensible reference model for data integration  
   G.N. Benadjajoud and B.T. David  206

PART FOUR  Information Infrastructure and Business Processes

19 Value System Redesign: system oriented management and enterprise integration in globally distributed manufacturing networks  
   B.R. Katzy and B. Riggers  221

20 Confidence in a network of cooperating companies – QM-documentation for certification  
   K. Mertins, M. Schwermer and F.-W. Jäkel  231

21 Applying information technology to minimise risks in satisfying organisational needs  
   O.J. Akomode, B. Lees and C. Irgens  242

22 Simulation supported business process performance assessment  
   M.M. Rajala  254

23 Key driven business reengineering – How to get Reengineering more efficient and effective  
   M. Stamminger  265

PART FIVE  Information Infrastructure and Engineering Processes

24 Generation of milling data in a virtual manufacturing framework  
   M. Matsuda and F. Kimura  277

25 Framework services for design data and design flow management  
   G.R. van Leuken and A. van der Hoeven  289

26 The concept of a design and planning platform  
   R. Atsumi  301

27 EXPRESS definition of vectorial tolerancing in product modelling  
   K. Martinsen and T. Kojima  313
PART SIX  Shop Floor Systems

28 Shop floor systems integration reference frameworks & life-cycle support tools
   J.J. Pinto Ferreira and J.M. Mendonça 327

29 Multiprotocol Ethemets on the shop-floor
   J. Pütter 335

30 Events in CIMOSA and the CCE platform
   S. Messina and P. Pleinevaux 344

31 Adaptable low cost shop-floor control system for Central and East European companies
   J. Uhl, Z. Smalec, J. Chrobot and J. Rakowski 356

PART SEVEN  Global Networks and Their Applications

32 Information infrastructure services for small and medium size manufacturers: The MI²CI project
   J. Goossenaerts and C.M. Acebedo 371

33 Environmental information systems based on physical flows
   A.I.D. Lambert, M.H. Jansen, R.V. Schuwer and M.A.M. Splinter 383

34 Strategies to develop a telemonitoring technology for machine tools via the World Wide Web
   I. Goncharenko, K. Mori and N. Kasashima 395

PART EIGHT  Towards Intelligent Manufacturing Systems

35 Globeman 21: Enterprise integration for global manufacturing towards the 21st Century
   Y. Kamio 409

36 Next Generation Manufacturing Systems in the IMS program
   T. Kurihara, J. Jordan and P.Bunce 418

37 Holonic manufacturing systems
   M. Höpf and C.F. Schaeffer 431

38 GNOSIS: Knowledge systematization; configuration systems for design and manufacturing
   M. Toyama 439

Index of contributors 453
Keyword index 455
Preface

On the verge of the global information society, enterprises are competing for markets that are becoming global and driven by customer demand, and where growing specialisation is pushing them to focus on core competencies and look for partnerships to provide products and services. Simultaneously the public demands environmentally sustainable industries and urges manufacturers to mind the whole life span of their products and production resources.

Information infrastructure systems are anticipated to offer services enabling and catalyzing the strategies of manufacturing companies responding to these challenges: they support the formation of extended enterprises, the mastering of full product and process life cycles, and the digitalization of the development process. Information infrastructure systems would accommodate access to and transformation of information as required by the various authorized stakeholders involved in the life phases of products or production resources. Services should be available to select and present all relevant information for situations involving any kind of players, during any life phase of a product or artifact, at any moment and at any place.

This book brings together a number of leading experts from around the world who have worked extensively on topics that matter for the design of information infrastructure systems for manufacturing. Experts from industry, consultants and researchers put their views forward on the current levels of industrial awareness, standards development, and research and development. Even though none of these themes is exhausted here, we believe the proceedings are essential reading for anyone wishing to further the design or development of information infrastructures for manufacturing industries.

Each of the chapters in this book are the result of presentations given by the authors at the second international conference on the Design of Information Infrastructure Systems for Manufacturing (DIISM'96), held in Kaatsheuvel, the Netherlands, September 15-19, 1996. The conference was supported by the International Federation of Information Processing (IFIP) through its Technical Committee 5 (Computer Applications in Technology) and its working groups 5.3 (Computer Aided Manufacturing) and 5.7 (Computer Applications in Production Management). Other sponsors were the Dutch PDI-CALS Center, the CIMOSA Association and the GLENnet Association. The conference was hosted by the Eindhoven University of Technology, Section Information and Technology, and BETA, the Netherlands Institute for Business Engineering and Technology Application.

The scientific part of the conference was followed by an industrial tour to two Dutch factories, NedCar in Born and Philips Medical Systems in Best, to study about their information infrastructure systems supporting customer-order driven production. We are grateful to our hosts at these factories.
We sincerely thank all authors, the programme committee members, and the conference participants for their contribution to the conference and this book. Special thanks goes to the members of the organizing committee, especially Michèl van Eekhout, Monique Jansen, Henk Jan Pels, Carla Schreurs and Arian Zwegers for their help in the preparation and administration of the conference.

We also gratefully mention the continuous encouragement that we received from Hiroyuki Yoshikawa, President of the University of Tokyo and Chairman of DIISM'93 (Tokyo, November 1993).

In conclusion, we strongly hope that this book will be a useful source of information for further research and development on information infrastructure systems for manufacturing, and that it may nurture ongoing research and development solving the problems and open issues highlighted during the conference.

Jan Goossenaerts
Fumihiko Kimura
Hans Wortmann

Eindhoven and Tokyo, December 1996
Introduction: Towards and information infrastructure for manufacturing industry

Emerging information and communications technologies (ICT) are adding new opportunities to enterprises, and changing the ways of competition and cooperation. On the verge of the global information society, enterprises are facing markets that are becoming global and driven by customer demand, and where competition is pushing them to focus on core competencies and look for partnerships to provide products and services. Simultaneously the public demands for environmentally sustainable industries are urging manufacturers to cope with the whole life span of products and production resources.

In the current transformation of industry, information and communications technologies (ICT) are playing an important role. Yet a coherent, clear vision is missing on how to achieve a durable connection between ICT and manufacturing and product life processes. The DIISM conferences (Design of Information Infrastructure Systems for Manufacturing) are dedicated to this theme. This introduction explains the concept “information infrastructure” and links it to some expected characteristics of future manufacturing industries. It concludes with some suggestions for further research which are drawn from the discussions at the DIISM’96 conference.

Characteristics of future manufacturing industries

Three fundamental characteristics are the emergence of the extended enterprise (or the network enterprise), the need to master the whole product and process life cycle, and the digitalization of the development process.

*Extended enterprises.* Cooperative arrangements in which several smaller companies come together to provide complex, customer defined products offer a number of advantages. In this way competition may exist for each core competence, for each component, production step or service. Moreover, because their use is not restricted to the product or service range of the (large) main manufacturer, resources such as expensive machine tools or expertise can be deployed for a wider range of products or services. Obsolete technologies and processes, and excess capacities can easily be identified and eliminated, and the competence portfolio -- the range of different core competencies and technologies mastered by an industry -- can grow more quickly. Given the increasing specialization of manufacturing industry (the use and recycling of more materials, the increasing precision and tooling needs of production processes, the rise of micro-electronics, software, mechatronic and intelligent systems) the growth of the competence portfolio in an industry is an important measure of its progress.

*Product life cycle.* The recent attention for the life cycles of products, services, processes and production resources originates from the increasing customer orientation, the waste problems, the prevention of technical accidents, and environmental pollution. An increasing number of parties, among whom are also public authorities which issue regulations on safety, pollution and recycling, influence the life phases of products and need access to the related data. Engineers have to include the full product life cycle in the development process. In the future more and more goods will circulate between the domains of production and consumption. The reduction of waste, and the improvement of product life cycles such that a larger amount of materials can be recycled, are important stepstones towards a sustainable industrial society.
The digitalization of the development process. Citing from the Final Report of the AIT Pilot Phase project: “The business process of the future will have two main phases. Firstly, a virtual phase, where the product, the manufacturing processes, and the manufacturing system will be designed and validated by means of computers and software. Secondly, the physical phase, where the actual realisation of the product is achieved”. The virtual phase of the development process (not to be confused with the “virtual organisation”) requires very demanding computing, reasoning and communication with data, information and knowledge on products, their components, production and product processes, resources, and business processes. Creating and working with product and process models requires a lot of time and expertise, and is also complicated by the need to simultaneously validate heterogeneous models that are typical for specific engineering disciplines (mechanics, electronics, material science, production engineering, business economics). Moreover different models are used during the successive phases of the development process. Ideally, models should be exchangeable, it should be easy to reuse them, and to perform joint analyses and validations, as for instance in a so-called digital mock-up. Harmonization should be performed across the boundaries of the different engineering disciplines and life phases of products and production resources.

ICT Infrastructure or Information Infrastructure?

ICTs will play an important role as enablers of these three trends in manufacturing industry, the integration of production and service processes of extended enterprises, the mastering of product and process life cycles, and the digitalization of the development process.

One should make a distinction between the ICT infrastructure and an information infrastructure. An ICT infrastructure results from the physical interconnection of various ICT components such that the exchange of data and communication between the components becomes possible. The existence of an ICT infrastructure is a necessary condition for creating and using an information infrastructure, but it is not a sufficient condition. From an information infrastructure we expect that it provides access to all information which is relevant and useful in a given situation, for any authorized user, during any life phase of a product, at any moment and at any place. On the one hand an information infrastructure should hide various application systems and the ICT infrastructure, on the other hand it should offer primitives enabling users to rapidly activate relevant information and applications supporting them in taking decisions or performing activities.

The Need for an Information Infrastructure

The future realisation of an information infrastructure will transform the handling of information and the productivity and innovativity in companies and society as much as the connection to water, transportation and energy infrastructures have done in the past, for households, manufacturing and distribution. Which role can an information infrastructure play as an enabler for extended enterprises, the mastering of product and process life cycles, and the digitalization of the development process?

Extended enterprises depend on information and communications technologies for sharing information and coordinating decisions and control. They need common methods and integrated telematics applications for concurrent engineering, contract negotiation, logistics and production, and for their further involvement in the product lives.

Approaching ICT application development on the basis of product families or extended enterprises, product family after product family, or extended enterprise after extended enterprise, may be acceptable for pilot developments, for demonstration purposes, and for building consensus about the required ICT enabled functionality. But in the long run such an
approach can not be viable. The problems of the islands of automation would simply be magnified at the level of inter-company cooperation across different product family life cycles. Also the independence and own development chances of each partner in the extended enterprise would be compromised because novel cooperative action must be preceded by a lengthy and expensive phase of mutual adjustment.

A harmonized approach crossing the various industrial sectors seems necessary if we want to exploit the strategic advantages of the extended and virtual enterprises. The availability of an information infrastructure concept for an industry in its totality could offer a lead. A large number of arrangements that are presently agreed (supply)-chain per chain or network per network, could be covered by an agreement covering the whole industry. This would allow partners in extended and virtual enterprises to even better focus on developing their core competencies.

Mastering the product- and process life cycle leads to additional requirements on the ICT applications that will connect companies, public authorities and citizens. In principle, each family of products may give rise to a class of services to cover the whole life cycle of the family and its occurrences: need evaluation, design and development, production and packaging, transportation, usage, maintenance, re-manufacturing, recycling and disposal.

All stakeholders should have the possibility to access relevant data and to update it, within their access-rights. This information will be distributed in space. It must be drawn from several parties. At the moment that the information is produced it is unpredictable when, where, by whom, and for which purpose the information will be used later on, nor where it will be stored in the future. The ownership of access-rights may change, as the ownership of goods. The information will also be distributed in time. Assembly information and part lists that have been defined during the development phase of a car must be accessible up to ten or more years after the production of a car, to support its disassembly. And the life-cycle models of the parts must be available for ensuring their proper processing. Data must be stored for very long periods of time, and be accessible at any moment, at any place, and for any authorized stakeholder.

The life-cycle orientation urges us to provide harmonized and stable solutions for the provision of product-related information in a society. An operational information infrastructure should keep globally available all potentially relevant data, in accordance with the prevailing access-rights.

Also the virtual development process relies heavily on an information infrastructure, for instance for exchanging specifications and digital mock-ups between sub-system suppliers and automotive or aerospace companies. We are still lacking modelling agreements which would enable us to join the partial models that originate from different engineering disciplines. Such agreements are required for an efficient and effective virtual phase of the development process.

An information infrastructure should ensure that models are exchangeable, easy to reuse, and allow joint analyses and validations. The possibility of achieving an information infrastructure emphasizes the need for a principle-based harmonization across the boundaries of the different engineering disciplines and life phases of products and production resources.

Other needs concern the property rights on digital mock-ups. These should be recognized and enforceable to avoid situations where digital mock-ups of components would be improperly used after employing them for analysis in the digital mock-up of a larger system.

**Unsolved Problems and Suggestions for further Research**

1. It is not clear who should realize the information infrastructure, and how it should be exploited. Who owns information, who pays for it, and what is paid for it? Given the wide scope of the problem and the anticipated large user community of the information
infrastructure it is not evident to identify the problem owner of this design and development problem.

2. The (scientific) world of enterprise modelling seems to be almost disconnected from the world of product modelling (such as in STEP) or product life-cycle modelling (such as pursued by the CALS initiative). It is recommended that these worlds try to learn from each other and be aware of each other’s existence and results for mutual benefit.

3. There is need for life-cycle modelling of more object classes than only (physical) products. More specifically, there is need for life-cycle modelling of resources, organizations, perhaps extended enterprises, software services, etc.

4. Many design and development efforts do not reflect a coherent vision on the future of manufacturing industry, and the role that ICTs will play in it. The rapid innovation in ICTs and software keeps the attention of specialists on the technology, at the expense of the possible deployment of the technology for achieving long term goals.

5. It is difficult to identify an ICT platform on which an information infrastructure can be built. As yet there is no “plug-and-play” concept for the joining of partial models of products and processes, and for their joint validation and analysis. Moreover, there is no generally accepted technology and know-how platform on which the infrastructure design and development can be successfully planned and implemented.

6. The increasing speed of technology development leads to a need for continuous education. If skilled engineers are supposed to be working for several decades in industry, it is clear that technology revolutions which take place every five years cannot be absorbed by industry without additional training and education. This is especially a problem in SMEs in countries with stable population.

7. Larger multinational companies that adopt available information and communication technologies provide insufficient feedback to the research and standardization communities. More feedback and communication can avoid the situation where the “wheel has to be invented over and over again”.

About this book

This book brings together a number of leading experts from around the world who have worked extensively on these and related topics. Experts from industry, consultants and researchers put their views forward on the current levels of industrial awareness, standards development, and research and development. These proceedings do not offer solutions for all the problems listed before, but it gives a fair overview of ongoing efforts and results on which to base further cooperative research and development. We hope that this book will be a useful source of information for further research and development on information infrastructures for manufacturing, and that it may nurture solutions for the open problems highlighted during the conference.

Jan Goossenaerts
Fumihiko Kimura
Hans Wortmann

Eindhoven and Tokyo, December 1996
Keynote and Invited Papers
Vehicle CALS
--A big challenge to virtual development

T. Mase
Nissan Motor Co., Ltd.
560-2, Okatsukoku, Atsugi City, Kanagawa, 243-01, Japan
Telephone: +81-462-70-1242
Facsimile: +81-462-70-1781
E-mail: mase@nova.lab.nissan.co.jp

Abstract
Japanese automobile manufacturers have maintained global competitiveness in the past by providing high quality, low cost vehicles. Recently, however, the strength of the yen and a recovering U.S. automobile industry has put the Japanese automobile export market in a precarious position.

Similarly, it is becoming apparent that, as information technology dramatically changes and advances, Japan lags behind in the field of key software and middle-ware technologies. In this environment, re-engineering of business processes in the automobile industry is becoming more important as a method to recover and maintain competitiveness in both the information and automobile industries as Japan moves towards the 21st century.

Consequently, the V-CALS consortium was established in May of this year by five major Japanese car manufactures, parts and die suppliers and other related companies. This enterprising project encompasses conformance testing to achieve virtual vehicle development while striving to merge competition with cooperation. This paper explains the aims of this consortium and gives an outline of the V-CALS project, focusing on the requirements for next-generation PDM, and trial use of STEP, EDI and SGML.

Keywords
Vehicle CALS, digital process, CAD/CAM, PDM, STEP, EDI, SGML

1 INTRODUCTION

Results for 1994 show that Japan's balance of trade in computer software was 259.5 billion yen (2.5 billion dollars) for imports and 13.5 billion for exports--a 19-fold deficit. Approximately three-quarters of the figure for software imports is represented by basic
software such as operating systems from the U.S. This is a strong indication that Japan's information industry is lagging behind the world market in general-purpose software. Similarly, there has been strong concern in recent years that the manufacturing industry in Japan is facing "de-industrialization." Specifically, if we look at the percentage of offshore production by the Japanese manufacturing industry in 1995, 80% of color televisions and 70% of VTRs were produced overseas. Predictions are now being made that, by the year 2000, 50% of Japanese vehicles will be produced in overseas countries.\(^1\)

Considering this situation, we believe that it will become more and more difficult for the traditional stalwarts of Japanese economic growth--the information and manufacturing industries--to continue in their present condition to sustain the Japanese economy and employment levels into the 21st century. To avoid further decline in the Japanese information industry, multifaceted policies are required. However, the role of the automobile industry should be based in its technical strength. But we do not want to be just an idle onlooker; we should rise to the challenge from our position in the automobile industry and make all-out efforts in the closing years of this century.

2 WHY IMPORT PERCENTAGES FOR SOFTWARE ARE SO HIGH IN JAPAN

There are a number of reasons for this situation. The first being that most software up until now has been developed for just one company. Many years of low mobility in the Japanese corporate market, spurred on by life-time employment, has led to a different job structure being established in each company. Inevitably what then happens is that a company's information systems department, being well-versed in the internal workings of the company, comes to play the lead role in software development. The job structure itself becomes to represent a company's know-how and competitive strength. It is only natural then that software is created to meet these needs, however, this software ends up lacking versatility and general-purpose qualities.

The second reason is that it is no longer feasible for a company's information systems department to continue to develop advanced, large scale software in this age of radical change in information technology. This change includes the shift from centralized to distributed systems with the advance of client server systems.

The third reason stems from the fact that the Japanese information industry lags behind the world trend in distributed systems and the use of commercial software. The forth reason is a somewhat more intrinsic problem. It can be said that Japan lacks the skills to develop creative software for use throughout the world. The final reason derives from the fact that software productivity is not significantly improving, as shown in Table 1, and there are considerable cost benefits in mass production of software. These reasons are helping to establish U.S. software as the de-facto standard, and as a result, pushing the market toward an oligopoly.

We believe that entirely new efforts in the area of information technology are required to sustain Japanese industry into the 21st century. As explained in this paper, it can also be said that new efforts are required to solve the issues presently confronting the Japanese automobile industry itself. A series of activities to solve these issues on an industry-wide level will no
doubt have timely and positive ramifications.

Table 1  Development Trends in the Information Processing Industry

<table>
<thead>
<tr>
<th></th>
<th>1955</th>
<th>1965</th>
<th>1975</th>
<th>1985</th>
</tr>
</thead>
<tbody>
<tr>
<td>Industry</td>
<td>1</td>
<td>20</td>
<td>80</td>
<td>320</td>
</tr>
<tr>
<td>Machine Performance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(hardware)</td>
<td>1</td>
<td>(10^2)</td>
<td>(10^4)</td>
<td>(10^6)</td>
</tr>
<tr>
<td>Programmer Productivity</td>
<td>1</td>
<td>2.4</td>
<td>5.6</td>
<td>13.3</td>
</tr>
<tr>
<td>System Reliability</td>
<td>1</td>
<td>5</td>
<td>24</td>
<td>120</td>
</tr>
</tbody>
</table>


3  ISSUES RELATED TO AUTOMOBILE DEVELOPMENT AND INFORMATION SYSTEMS

3.1  Present status of CAD/CAM in the automobile industry and related issues--focusing on the situation at Nissan

The automobile industry has been working towards practical application of CAD/CAM. At Nissan Motors, we have pushed ahead with computer applications in a variety of vehicle development and production preparation fields, including styling design and die processing for body development. We have also constructed a progressive job structure using digital information. These efforts have greatly contributed to shorter development periods and improved quality. The outcome of this being an increase in the exchange of CAD data between Nissan and its overseas offices, and related manufacturers. However, exchanges between different systems have not always been successful, causing considerable problems in recent years. In response to this problem, a working group to standardize exchange of CAD data was established three years ago within the Japan Automobile Manufacturers Association, also know as JAMA, in an effort to at least perform correct data exchange between automobile-related companies. This group has studied IGES and STEP exchange standards. Through these activities, work has nearly been completed on an industry-wide interpretation of IGES standards, as well as the development and modification of data exchange functions for commercial and individually-developed software based on this interpretation. Presently, focus has shifted to technical studies on STEP exchange standards, and with the cooperation of Europe and the United States, work is centering on AP214 with the aim of early standardization of the data required in the vehicle development process.

3.2  Demand for a shorter development period for new vehicles

Another major issue facing the automobile industry is how to respond to the strong demand for a reduction in the time it takes to develop a new vehicle. During the 1980's, the Japanese automobile industry required an average of two-and-a-half years from model approval to commencement of production. However, in the last couple of years, intense competition and
drastic changes in the market have resulted in strong calls for a reduction in this development period, with new vehicles presently being developed within a period of twenty months. This reduction has done little to quell demands, and intense competition in the Japanese domestic market has renewed calls for an even further reduction in development time. To answer these calls, it has become essential to push ahead with concurrent engineering, as well as use “digital processes” to develop new vehicles.

Not only is it important that the information industry builds up competitive strength in the international market, it is vital for the automobile industry that the development process is digitized to the highest level in order to support concurrent engineering, as well as the more effective use of CAD/CAM/CAE as essential tools in the development of new vehicles.

4 ESTABLISHMENT OF THE CALS RESEARCH PARTNERSHIP

With the conclusion of the Cold War, the U.S. government has changed its technical policies away from defense research and development to the development of advanced technology for social welfare. For many years the U.S. has been the overwhelming world leader in the field of advanced computers and communications, even after focus shifted to social welfare projects. Europe and the EU are also conducting activities to strengthen infrastructure for industrial technology. The Japanese Ministry of International Trade and Industry, fearing that Japan would fall further behind the rest of the world, established Nippon CALS Research Partnership (NCALS) in July of last year with the cooperation of the private sector. Initially it was decided that this consortium would target fields related to electric power industry, and aim to conduct conformance research on Electronic Commerce (EC). At the same time, a number of industries were preparing the groundwork on which to conduct similar research, and in May this year, it was decided that the Vehicles CALS Consortium, V-CALS, would be established by five major Japanese automobile manufacturers with the cooperation of die and other related suppliers, and information companies. V-CALS will attempt to solve the issues that are inherent to the automobile industry, as well as formulate ways to deal with the problems confronting the information and manufacturing industries in Japan.

5 OUTLINE OF V-CALS

5.1 Aims

V-CALS aims to achieve a digital process, encompassing new vehicle development through to production preparation, whereby development is conducted using digital data such as CAD data and documents, instead of approving designs on the basis of hand-written blueprints and materials, or creating prototypes to investigate design. For this reason, the following activities will be conducted:

1) clarification of prevailing problems and issues;
2) clarification of the common information system platform specifications and the essential requirements to solve these problems, and conduct conformance research on next-generation information systems and new job structures for the fields concerned.(2)
5.2 Construction of V-CALS

Four working groups (WG1–WG4) will be set-up and the following conformance research conducted.

- Testing will be conducted on a digital process using the technology presently available, and requirements will be clarified.
- Research will be conducted on the tools (PDM) required to achieve the next-generation digital process for future activities.

Working group two to working group four will establish the standards that will form the foundations of the digital process, and will development and assess translators. More specifically, working group two will put vehicle application protocol AP214 to practical use in STEP; working group three will research ways to practically apply EDI, namely the automobile industry standard EDI for receiving and placing parts orders for mass production activities; while working group four will prepare automotive maintenance manuals using SGML, and database information on regulations in each country.

Figure 1 shows the connection between all the related functions of these working groups. Likewise, Figure 2 shows the structure of the V-CALS Consortium.

![Figure 1 Functions of V-CALS](image-url)
5.3 The activities of each working group

1) Working group one
This group is divided into two subgroups (SG11, SG12).

<SG11>
This subgroup will conduct testing on digitalization of products and process management, as well as concurrent development using the most sophisticated information systems available. This testing will substantiate the feasibility of digital processes, and highlight the problems and issues associated with the CALS technology required to achieve these digital processes.

<SG12>
This group will research next-generation PDM or product data management. PDM here means the set-up whereby a database manages all the data required for product development, including CAD data, specifications, parts lists and technical data. The PDM requirements in the automobile industry are virtually limitless, and include:

- a huge number of parts, somewhere in the millions, and a massive volume of information used when assembling these parts on a solid model;
- a huge number of companies and people related to this industry;
- and the development and production activities being conducted on a worldwide scale.

This subgroup will develop a next-generation PDM that can handle all these requirements, and determine the appropriate information system platform.

2) Working group two
This group will

- support efforts to standardize STEP;
- research actual installation of STEP standards, and conduct conformance testing;
- and examine application rules for product model data.

These activities will focus especially on AP214.

3) Working group three
This group aims to create an electronic parts procurement system, and push ahead with the practical application of the JAMA-EDI standard already established by JAMA. To achieve
these aims, industry-wide support software will be developed, and problems will be amended by way of conformance testing.

4) Working group four
This working group is also divided into two subgroups (SG41, SG42).

<SG41>
This subgroup aims to standardize data and make it more "open" through use of SGML in an effort to create electronic service manuals. Attempts to standardize the information contained in service manuals of the five major domestic automobile manufacturers will enable automotive servicing companies and affiliated dealers to retrieve and refer to this information via public circuit networks or the Internet. This subgroup will determine the information system required to achieve this, and clarify the operational issues that must be solved.

<SG42>
This subgroup conducts conformance testing on construction and practical use of vehicle regulation database. In this test, world main regulations (MVSS, ECE, EC, ADR, etc.) are stored in both Japanese and English languages using SGML format. This subgroup will clarify the issues on construction, utilization and operation of regulation database, for the purpose of easier and more efficient exchange of information.

5.4 Results

The following results can be anticipated from the activities to be conducted in this project.

1) The following will be the returns shared with the manufacturing industry:

- common tools for STEP/AP203 translator;
- know-how on applying CALS technology to the process of creating 3D structures;
- and process management tools and technology common to the automobile and manufacturing industries.

2) The following are the returns specific to the automobile industry:

- know-how on application technology for the development/production preparation process using CALS technology;
- technical data for refining ISO standards, including STEP/AP214;
- technology for the preparation of electronic service manuals, parts catalogs and other materials specific to automobiles;
- and know-how and basic technology for establishing a database network operation center.

5.5 Schedule

The first phase of conformance research on the digital process will be conducted until the end of March 1998, with efforts to revolve around development of element technologies. Based on the results of this research, the second phase--development of required technologies--will
be commenced from April 1998.

6 CONCLUSION

V-CALS will attempt to reduce the technical problems associated with information systems and computers, and work out ways to reform work methods by presenting the requirements necessary to achieve a digital process, as well as extract measures to fulfill these requirements through virtual development and conformance testing of virtual companies. Through these activities, we would like to contribute to:

- the development of new middle-ware for use throughout the world from the year 2010;
- help sustain the competitiveness of the automobile industry;
- and collaborate with Europe & the U.S on the creation of common platforms and standardization efforts.

The consortium brings together a variety of different corporations. As a result, there is inherent difficulties with tackling common issues. However, we have come to the point where we must do something more than just compete. It is considered that activities that successfully combine competition with cooperation are of the utmost importance to the fields mentioned here today. However, we are also convinced that these activities are not only important to Japan in the 21st century, but also to the environmental preservation of the shared resources of this earth, and the well-being of all mankind.

The combined strength of many people will make this project a big success.

7 REFERENCES

1. Structural Reform of Technology and the Japanese Economy, Shiro Kurihara, Business Review Vol.43, No.4

8 BIOGRAPHY

Toshiaki Mase received the B.S degree in mechanical engineering from Kyoto University. He was employed by Nissan Motor Co. LTD. in 1967. He has been working in the area of computer technology and engineering systems development for more than 25 years. He managed the projects for developing Nissan and parts venders CAD/CAM systems, database systems and styling CAD/CAM system. He was senior manager of body design department in 1990. He is currently general manager of Engineering Systems Department Information Systems Division. He is chairman of V-CALS consortium steering committee.
Changes in the industrial environment and CALS

L. Zuijdegeest
PDI/CALS Centre
P.O. Box 770
3430 AT Nieuwegein, The Netherlands
Phone: +31 (0)30 6021551; Fax: +31 (0)30 6022469
E-mail: calsnl@euronet.nl

Abstract
CALS is a management strategy for the implementation of modern working methods supported by information technology based on widely accepted standards. Modern working methods are based on early involvement of all relevant functions. Product data should be available for all concerned. The information technology and the necessary standards will be increasingly model-based rather than document based. CALS has international support from all industry sectors.

Keywords
CALS, information technology, STEP, concurrent engineering, logistic support, SGML

1 INTRODUCTION

In the past few decades, industry has seen quite a few changes. Never, it seems, has the pace of change been so high as right now. Major companies, believed to be there forever, came close to extinction. The ultimate punishment for not adapting to change. Nearly always management points to one or more reasons factors beyond control to indicate that the company itself is not to blame.

Despite all best intentions, most companies, large and small, have suffered great hardship in the past decade. Some have survived and have been strengthened by the experience, some are still suffering and others did not survive. Examples are well known, also in the Eindhoven region, the industrial heart of Holland.
Changes in the industrial environment not only present threats, but also opportunities. When seized, opportunities can lead to differentiation from competitors and increased value for its owners.
Currently industry faces an number of changes. Changes that require a new of thinking on how to conduct business and a new way of thinking on the use of information technology.

2 CHANGES IN THE INDUSTRIAL ENVIRONMENT

System or product complexity
In the past centuries, scientific and technological developments have been successful by man’s ability to reduce systems to easy understandable and manageable components. This approach has resulted in many highly specialised functions, often organised in a Taylor-like fashion leading to ‘functional islands’.
In the past decade or two, the disadvantages of this approach have become more and more obvious.
With increasing complexity of products and systems, it is clear that their behaviour cannot be understood and managed by its components alone. The relationships between components are often more important for the behaviour of systems than the components.
In designing new systems or products, more attention should be paid to the relations between components. This means that the functional islands have to work closely together, if not integrate.

Focus on client’s business
Selling products in today’s market requires an in depth feel for the clients business. In a recent study conducted in Dutch industry, the overall conclusion was that time-to-market is the factor that counted most in purchasing products. Product cost came next and product quality came last of these three. An additional factor that gains importance is cost-of-ownership or life-cycle cost. Although well-known in the aerospace and defence-business, the factor is less appreciated elsewhere. Mostly because the purchasing party sees initial expense and operation cost independent of each other. Literature shows the following issues in different industry sectors (table 1).

<table>
<thead>
<tr>
<th>Consumer Products</th>
<th>Industrial Products</th>
<th>Industrial Projects</th>
<th>Infrastructure projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduce time to market</td>
<td>Reduce cost</td>
<td>Reduce proposal lead-time</td>
<td>Meet quality</td>
</tr>
<tr>
<td>Continuous reduce cost</td>
<td>Reduce lead-time</td>
<td>Reduce project cost</td>
<td>Reduce cost</td>
</tr>
<tr>
<td>Continuous improve quality</td>
<td>Improve quality</td>
<td>Realise set of functional standards</td>
<td>Manage and reduce lead-time</td>
</tr>
<tr>
<td>e.g. telephone</td>
<td>e.g. car</td>
<td>e.g. ship</td>
<td>e.g. bridge</td>
</tr>
</tbody>
</table>
Every business-function has its effect on other functions and a clear overall insight is necessary to reduce development and production time, keep initial and life-cycle cost low and achieve acceptable quality.

**Concentrate on core business**
Also in the past decade, companies felt the need to concentrate on core-business and sold-off many of their non-core-business activities. Former departments became companies in their own right or were taken over by other companies and offered their services to the previous owner.

**Business-unit structure**
In-house the situation changed even more. Business-unit structures were introduced and the business-units adopted a free market approach and obtained the freedom to hire services from other business-units or from outside the company. When combined with the increasing product complexity and the need to integrate functions, companies now face the challenge to get both internal business-units and external contractors working closely together on product development and manufacture.

**Subcontracting**
The trend to focus on core business and the trend towards independent business units implies the wish to make full use of the capabilities of subcontractors. Until recently, end manufacturers thought it necessary to make detailed specifications for contractors. These contractors only supply to specification, no other input is required from them. The specification depth is rapidly decreasing. The end manufacturer provides functional specifications and boundary conditions when applicable, and the contractor goes to work, using his full expertise. He thus not only saves the end manufacturer the cost of having the expertise in house, but also creates added value for his own business.

**Virtual company**
All these trends signify an increase in the importance of the supplier. The structure in the industry starts to look like a network of more or less equivalent partners, rather than the traditional chain of suppliers.

**Concurrent engineering**
Time to market, initial cost, cost of ownership and product quality are, often in this order, the key elements on which a company or an individual purchases a product. In many industrial sectors, the phenomenon of concurrent engineering is seen as the way to improve on these key elements. Within companies serious efforts have been made to achieve this concurrent, multidisciplinary approach. But with the increasing importance of suppliers, an internal concurrence will not suffice. Essentially the suppliers are still sequentially involved. Given the potential contributions suppliers can make to effectivity and efficiency, a more involving working method is necessary. The concurrence should also involve the main suppliers: cooperative engineering (figure 1).
Figure 1  Decreasing product development time with co-operative engineering

Quality of management
Business functions need to work closely together. Multidisciplinary teams are formed to integrate functions and get access to knowledge in all relevant departments. Many companies that have tried to form multidisciplinary teams had difficulty implementing them. To get a group of people from different backgrounds communicating is quite difficult to achieve in practice.

New organisations and working methods require a new kind of management. The traditional power-base of the middle manager is under threat. In order to make co-operative engineering work, new views on conducting a business are necessary. An end manufacturer and his suppliers can no longer be seen as a group of individual companies. The group of companies should function as a virtual enterprise. The virtual enterprise can only function when departmental thinking no longer exists. In multidisciplinary teams information should be exchanged freely and responsibility should be shared by the participating disciplines.

3 ADAPTING TO CHANGE

When looking at the changes in industry there is not one simple way for companies to adapt. Companies must realise that the new way of conducting their business has to be based on the timely involvement of experts. Already in the conceptual stages of product development a great part of the product’s life-cycle properties, will be determined. Properties include manufacturability, manufacturing cost, maintainability and maintenance cost, quality, re-use and so on. So in the stages of conceptual design and engineering, companies should also work on production planning, logistic support and technical and training documentation.
To achieve this, processes throughout the company will have to be reviewed and revised. For instance, on the one hand engineering is often seen as a cost factor with rather low investments. The manufacturing department on the other hand gets great attention in order to optimise production process. The manufacturing department heavily depends on the quality of the design. Although this has been known for a long time, both industry and government are slow to adapt.

Adapting to change would mean changing business processes, changing the role of people, making use of widely accepted standards (both functional and technical) and use modern information technology based on these standards.

A coherent overall strategy is necessary involving business processes, people, information and (information) technology. CALS is such a strategy (figure 2).

---

**Figure 2**  CALS: a coherent overall strategy involving business processes, people, information and (information) technology

4  INTRODUCING CALS

4.1  Where did CALS come from?

CALS started life on the morning of 24 September 1985 when William H. Taft Issued a memorandum ‘Subject: Computer Aided Logistic Support’. This memorandum from the US Deputy Secretary of Defence to the secretaries of all US military departments and directors of defence agencies outlined a strategy for major improvements in supportable weapon system
designs, and in the accuracy, timeliness and use of logistic technical information. This technical information includes technical manuals, training material and product definition data.

Taft launched another memorandum on 5 August 1998 in which CALS is expanded to cover the acquisition process: Computer-aided Acquisition and Logistic Support. This memorandum demanded the use of national and international standards for delivery, access and management of digital data.

CALS is now defined as Continuous Acquisition and Life-cycle Support, shedding its image of a technically oriented strategy and emphasising the life-cycle concept. The CALS Strategy is now globally accepted as 'the development of an integrated data environment created by applying the best commercial technologies, processes and standards for the development, management, exchange and use of business and technical information among government and industry enterprises'. Or in other words, CALS is a management strategy to improve business processes and make better use of the most important production factor, information.

CALS now has global support, in the United States, in Europe and in the Pacific Rim in both defence and non-defence industry. Supporting industry bodies have been founded in the past few years to discuss progress, CALS-programs and projects, and experiences.

4.2 Harmonising Business Processes, Standards and Information Technology

The technological basis of the virtual enterprise lies in information technology. Organisational and procedural aspects are by far the most important of the virtual enterprise. Information technology is the enabling technology. Without it, the virtual enterprise would be impossible to achieve.

The virtual enterprise and information technology based on standards is what the CALS strategy is all about. CALS, Continuous Acquisition and Life-cycle Support, originates from an industry sector where many companies are involved with the development and production of very complex capital goods that have to meet high demands in product quality and maintainability.

For the business processes to function, access to information by all concerned is necessary. This implies an information infrastructure in which data can be generated, stored, managed, retrieved and archived. Parties concerned should have access based on their roles and responsibilities in the business processes.

4.3 CALS Phases

The CALS strategy is designed to be implemented in two steps (figure 3). The first step, which has been more or less completed, is focused on the migration from a paper based information flow to a digital information flow. In the current phase the exchange of data between
applications is based on neutral (industry) standards. Edifact and Iges being examples. This is an interim solution.

The second step is aimed at the migration from the interim solution towards integrated product model databases. In the defence-industry, CALS is based on the CITIS-concept. CITIS is the acronym for Contractor Integrated Technical Information Services and is intended to provide government agencies with real-time, on-line access to technical information that exists and is maintained at the contractor site. In fact, government agencies have outsourced the management of weapon systems to the prime contractor.

CITIS provides access to design data, data for logistic support, training manuals and maintenance manuals. For each of these data types, separate functional and technical standards are prescribed.

![Image: CALS phases](image)

---

**Figure 3** CALS phases

Ideally, these standards should provide the possibility create data models based on activity models. This way, the information infrastructure meets the requirements of the business process.

The most prominent standards are:

- **SGML**: ISO 8879 and related standards like HyTime for modelling textual information and linking objects for hypermedia purposes;
- **LSA(R)**: MIL-STD-1388-1A and 2B functional and technical standards for logistic support analysis and modelling logistic support databases;
- **STEP**: ISO 10303 functional and technical standard for modelling product data.

Due to the severe budget reductions, governments have been looking for ways to cut defence expenses. The cost to maintain specific military standards for the defence industry is so high, that Secretary of Defence Perry issued the Perry-memorandum in which he announced the migration towards internationally accepted standards and more off-the-shelf products.
For instance, the LSA(R)-standards are currently harmonised for the NH90-project and a first draft will be submitted to ISO for further development and balloting.

4.4 STEP

STEP has world-wide support. The real value of STEP is that it is both an architecture and a standard. It supplies the methods to develop activity and data models and the implementation of product databases for managing, sharing and archiving product data throughout the product’s life-cycle. It implies a separation of functionality of computer applications and data representation. Ultimately, computer applications should read/write directly in STEP databases.

4.5 STEP drawbacks

Since its conception in 1984 a lot of effort has been put into the development of STEP. The results however, have not lived up to the expectations. Partly this is due to the ISO standardisation process, and partly due to the structure and complexity of STEP.

The basis of STEP originally was a single data model, called IPIM (Integrated Product Information Model) that encompasses all industry sectors, all disciplines and all types of information. The first version of IPIM was released in 1988 and consisted of hundreds of pages, while only covering a small portion of its scope. It would be impossible to cope with such a modelling task.

A more pragmatic view was adopted with Application Protocols (APs). An AP contains a data model for a particular type of data and a particular industry sector. Although more sensible than the IPIM, APs still have large scopes that require extensive data modelling and a lot of time. The Application Protocol for the automotive industry is a prime example. Its scope is so large that consensus might be difficult to attain.

Furthermore, APs must be used in conjunction with each other and therefore be interoperable. They are not. Simply because the industry specific terminology is used without reference to other APs.

This problem with interoperability led to yet another approach, based on the enterprise rather than on the industry sector. Small standardised STEP-components called Application Interpreted Constructs (AICs), provide building blocks for STEP product models. The drawback is the expected volume of these building blocks.

On the other hand, companies can tailor their information needs more precisely to their processes. In order to do so, software tools must be available to facilitate this tailoring process. This might lead to decreased interoperability but again availability of tools should be sufficient to counter this potential problem.
5 CONCLUSIONS AND FUTURE

Changes in the industrial environment create the need for a coherent strategy on processes, people, standards and information technology. Not one of these factors will do without the others. Standards are becoming increasingly available to support processes, although not without problems. Companies should focus first and foremost on their business processes. Only then should they start working on their information infrastructure. Information technology should ideally be based on widely accepted standards. When standards are not yet mature, pragmatism should take over and implementation should be based on what is available. Under no circumstances the absence of matured standards can be an excuse for not adapting to change.

6 REFERENCES

In the composition of this paper the author used various sources amongst which:
• Engineering Data Newsletter (ed. K. Al-Timimi)
• Presentation ‘From supplier chain to supplier network’ by W.F. Gielingh at the PDI/CALS Conference 1995
• CALS Handbook by PDI/CALS Centre
• Introductory Course on CALS by PDI/CALS Centre

7 BIOGRAPHY

L. (Bertus) Zuijdgeest MSc. founded with others in 1992 the PDI/CALS Centre of which he is currently managing director. He is involved in many activities to increase the awareness and knowledge in Dutch industry in modern IT and standards, amongst which STEP-projects in the process and automotive industry. He also is CAD-editor for CA Techniek, the leading CAD/CAM/CAE-magazine in the Netherlands. Previously he worked as project manager on projects involving CIM and artificial intelligence.
Abstract

The world wide demand for integrated CAD/CAM/CAE software solutions is growing rapidly. The computer professional services business which includes systems integrators is expected to grow to meet that demand because data exchange formats and interface protocols between CAD/CAM/CAE software applications are for the most part incompatible. This incompatibility results in increased business opportunities for professional integrators who create custom integration solutions to support data exchange between software applications. These solutions are expensive to implement, require a great deal of time to develop, are very inflexible, and are not based upon industry standards. These custom solutions result in numerous problems that can be broken down into three major technical impediments: 1) the lack of understanding of what information and knowledge is shared between manufacturing design, planning, and production areas, 2) the lack of information standards that define structure and content of data that must be shared by multiple manufacturing applications, and 3) the lack of standard interface protocols between support systems, e.g., communications and database management systems, that would facilitate the sharing of information between independently-developed software applications.

This paper presents perspectives on engineering tool integration issues. It also describes work underway to address those issues at NIST in Computer-Aided Manufacturing Engineering (CAME) and Systems Integration for Manufacturing Applications (SIMA) programs. Some of the technical activities include integration of design, process planning, plant layout, scheduling, and production simulation systems. A virtual production facility has been established using simulation and virtual reality systems that will provide a basis for validating manufacturing data before it is released to the shop floor.

Keywords
Manufacturing software integration, manufacturing engineering, simulation, virtual manufacturing, process modeling
1 INTRODUCTION

Just as computer-aided design and engineering tools have revolutionized product design during the past decade, computer-based tools for production system engineering could revolutionize manufacturing. The major problem today is the lack of software integration—engineers need to move data between tools in a common computing environment. A recent NIST study of engineering tools has identified more than 400 engineering software products marketed today, most all of which are virtually incompatible with one another. That is, interoperability between these tools is for the most part, non-existent.

Tool kit environments are needed which integrate clusters of functions that manufacturing engineers need in order to perform related sets of tasks. Some examples of different types of tool kit environments which are needed by manufacturing engineers include:

- **Manufacturing Engineering Tool Kit** - Used to develop process plans for machined parts, identify manufacturing resource requirements from product design data, and validate those plans using simulation technology.
- **Production System Engineering Tool Kit** - Used to translate product design, production demand forecasts, and other constraints into a production system design. Also provides tools for evaluating the performance of candidate designs.
- **Business Process Re-engineering and Producibility Analysis Tool Kit** - Used to analyze and re-design business processes associated with manufacturing and evaluate the product producibility, i.e., the relative ease by which a process may be produced.
- **Quality Engineering Tool Kit** - Used to develop quality models and metrics for processes, gather and evaluate statistical process control data, and initiating fine tuning of process parameters.

If these tool kit environments were available today, they would not only be used by manufacturing engineers, but also by product designers, quality engineers, industrial engineers, system engineers, process planners, tool designers, and manufacturing management. In the future, new users may include: manufacturing strategists, producibility engineers, enterprise designers, benchmarking teams, etc.

1.1 NIST Programs in Manufacturing Software Integration

The NIST Systems Integration for Manufacturing Applications (SIMA) Program, Barkmeyer (1995) and the NIST/Navy Computer-Aided Manufacturing Engineering (CAME) Program, McLean (1993) are focusing on providing the models, frameworks, operating environment, common databases, and interface standards for integrating a wide variety of tools for designing manufacturing processes, equipment, and enterprises.

In collaboration with industry, the CAME program is assessing needs with respect to manufacturing engineering tool integration. The program is developing architectures, interfaces, and databases for integrating engineering tools environments. Prototype integrated engineering tool kits are also being constructed using commercial products to validate interface specifications. Integrated tool kit solutions will be evaluated at industry sites. The principal elements of the program's technical approach are:

1) Identify and address critical industrial needs through collaboration,
2) Develop solutions to engineering tool integration problems,
3) Construct prototype environments using commercial products,
4) Validate results through industrial testing of system implementations,
5) Specify and promote needed industry standards, and
6) Facilitate the rapid commercialization of new technology.

A study conducted by the U.S. Department of Defense suggested that industry can obtain major economic benefits from increased investment in: 1) integration methodologies, 2) simulation and modeling, and 3) manufacturing/industrial engineering support tools. Some of the benefits obtained from the CAME program are outlined below.

**Improved engineering function**
Engineering tool kit technology will help manufacturing engineers make better decisions and more quickly evaluate the effects of those decisions. By improving process planning and simulation capabilities, a much greater percentage of products will be produced correctly the first time. Furthermore, the overall time to perform the engineering function will also be reduced if fewer changes to plans and programs are required once a job hits the shop floor. These improvements will result in fewer scrapped parts and less re-work. The integration of software packages and common databases will ensure that less time is wasted re-entering the same data into multiple engineering tools.

**General productivity benefits**
A number of broader benefits will be seen as a result of improvements in the manufacturing engineering function. There will be better utilization of shop floor equipment. Shops will be able to respond quicker to rush orders if their resources are used more efficiently. More energy can be devoted to producing higher quality products. Shorter response times will be realized for spare/repair parts for existing products as well as new products.

**Increased availability of engineering products**
Finally, the cost of performing the engineering function will be reduced if commercial software and hardware can be made more accessible to a larger group of users. This project will help increase the interoperability and value added by engineering tools. It is likely that this will lead to a greater demand and market for these commercial products. As the sales of these products increase, the "per unit" costs of acquiring software at new sites should be expected to go down.

**1.2 Overview of Paper**

The remainder of this paper describes the work underway at NIST using one of several engineering tool integration projects as an example of our technical approach. The project is focusing on developing interfaces for integrating tools for production system engineering. Section 2 of the paper provides an overview of the top two levels of the process model for production system engineering which has been developed to guide tool integration. Section 3 describes the integrated manufacturing engineering tools environment, the virtual manufacturing system which is being used to test tool integration today, and outlines future work.

**2 PRODUCTION SYSTEM ENGINEERING PROCESS MODEL**

The Production System Engineering Tool Kit under development by NIST and collaborators will provide functions to specify, design, engineer, simulate, analyze, and evaluate a production system.
Other functions included within the environment are project management and budgeting. Examples of production systems which may eventually be engineered using this environment include: transfer lines, group technology cells, automated or manually-operated workstations, customized multi-purpose equipment, and entire plants. The initial focus for this project is on small production lines used to assemble power tools. A process model is being used to define interfaces which are needed to integrate the tool kit.

A process model is one of several models that are needed to implement an integrated engineering tools environment. The process model defines the functions that tools must perform in order to engineer a production system. The model also defines inputs, outputs, controls, and mechanisms for carrying out the functions. The process model is a key reference document for defining the data flows and interfaces between the modules in the integrated environment.

The process model for production system engineering has been developed using Integrated Definition Method (IDEF0) modeling techniques and the Meta Software Design/IDEF™ tool, see Meta (1994). The model defines the tool kit functions and data inputs/outputs for each function. Detailed information models are under development which further specify each data input and output identified in the process model. The information models are being used to implement shared databases, exchange files, messages, and program calls for passing information between the commercial software tools.

The zero level of the model identifies the production system engineering function, its inputs, and its outputs. The first level of the model decomposes the engineering process into five major functions or activities: 1) define the production system engineering problem, 2) specify production processes required to produce the product, 3) design the production system, 4) model the system using simulation and evaluate its performance under expected operating conditions, and 5) prepare plans and budgets. Inputs to the production system engineering function include: production requirements, product specifications, quality, time, and cost constraints, and manufacturing resources. Outputs of the function include: process specifications, simulation models, performance analyses, system specifications, implementation plans, and budgets.

Figures 1 and 2 illustrate the first two levels of the IDEF0 model. The model further decomposes each of these functions and data flows into sub-levels. Brief summaries of the sub-levels are presented in the sections that follow.

![Diagram](image)

**Figure 1** Top level of the production system engineering IDEF model.
2.1 Engineering Problem Definition

The first step in engineering the production system is clearly identifying the production engineering problem which is to be solved. Problem definition data will influence how all of the other production engineering functions are carried out. This activity is primarily one of gathering and organizing data from a number of different sources. Ultimately, data gathered as a part of this activity would be recorded in template forms, imported from other applications, and maintained in a shared database. Critical data which must be identified to initiate the engineering process includes:

- Product data and key product attributes - product name, part number, model number, description, functionality, product structure (bill of materials), material composition, dimensions, weight, reference drawings, part geometry models, part family or group technology classification codes, technical specifications, reference documents,
- Production system and engineering project type - new production system (e.g., plant, line, cell), modification to existing system (i.e., product or process changes), relocation of system to new site, phasing out of a production system,

![Figure 2](First level of decomposition of the production system engineering model.)

- Manufacturing constraints and issues - market forecast and production rates required (minimum, normal and peak production rates in units/hour, units/shift, units/day, units/year), production capacity, level of automation versus manual operation expected, information and control system requirements, target production site(s), floor space limitations, quality and yield requirements, safety stock requirements, storage availability, known environmental or safety hazards, production plant calendar,
- Critical milestone dates and schedules - production ramp up plan, target dates for: system requirements specified, system design completed, requests-for-proposals issued, systems installed, testing completed, training completed, system operational, post production support, system phase out,
- Expected or estimated costs - product price, manufacturing cost, system implementation, operating costs,
Manufacturing data for related products - production engineering data for this or previously manufactured products (in some cases all outputs from previous engineering projects), competitor products and sites, possible benchmarking sites.

With the exception of critical milestone dates, most of the information outlined above may at some point be used by the next function in the process model, i.e., the specification of production and support processes. All data may be used directly by other downstream functions, if appropriate. During the course of the production system engineering process, downstream functions may provide feedback suggesting changes to the problem definition data.

2.2 Production And Support Process Specification

The second phase of the production system engineering activity is to develop a process specification for the production and support operations required to manufacture the product, see Tanner (1985), Salvendy (1992), and Sule (1994). Data developed during this phase will ultimately take the form of directed graphs and/or flowcharts. Nodes in the graphs will contain attributes which identify processes and their parameters.

A manufacturing/assembly precedence structure diagram is developed from the product geometry data and bill of materials. From the precedence structure, processes and processing precedence constraints may be derived. The derivation process may be based on human experience and intelligence, or implemented as a rule-based expert system. Data developed by this function includes:

- Process identification - process name, process type (operation, storage, inspection, delay, transportation, information, or combined activity), process parameters,
- Process resources - input product components, output product (subassembly or part identifier), tooling and fixtures, staff and job skill requirements, process by-products and hazards,
- Process time and costs - process duration, estimated process cost, product value-added.

This process is recursive--high level processes are decomposed into sub-processes until all basic or primitive operations are specified. Constraints on groups of processes and operations are identified and precedence relationships are specified.

Process specifications are perhaps best represented as diagrams and/or tables. Graphical editing functions and human interaction are normally required to layout diagrams in an understandable form. Large diagrams may be unwieldy and should be decomposed into multiple levels of sub-diagrams.

Other process specification data which may be developed as part of this phase include:

- activity relationship matrices are defined which describe how different processes relate to each other, e.g., required proximity or location.
- specification of requirements for processes, tooling, job skills, timing and line balancing, quality control, process audits,
- development of process and inspection plans, process description sheets,
- development of time standards for operations,
- ergonomic analyses of manual tasks,
- value engineering analysis (i.e., determination of job activities/steps which can be eliminated).

Processing scenarios may also be defined which describe how production will be carried out before, during, and after the new production system is implemented.
Process specifications next must be reviewed and revised to correct errors, inconsistencies, etc. Feedback requesting changes to the problem definition, as the process specification is developed. As the system design is developed in the next phase, feedback may be provided indicating required changes in process specifications.

### 2.3 Production System Design

The third phase of the engineering process is production system design. This activity includes the design of the physical processing systems, material storage and delivery systems, and information management/control systems for the production system. The production system design problem is addressed in Sule (1994). The mechanical assembly system and flexible manufacturing system problems are described, respectively in Nevins (1989) and Draper (1984). Facility layout is presented in Apple (1977) and Francis (1992). Manufacturing system architecture, design, and specification development processes are defined in Rechtin (1991), Bertain (1987), Rembold (1993), Compton (1988), and Purdy (1991).

A generic decomposition of production system design is: 1) define system requirements for each process, 2) assign requirements to system modules, 3) develop system operating scenarios for the modules, 3) identify candidate systems/machines/tooling for each module, 4) evaluate alternative technologies and candidate offerings, 5) determine number of systems required based on processing cycle time and required throughput, 6) conduct system build or buy analyses, 7) select systems for acquisition, and 8) developed detailed design for overall system based upon build and buy decisions.

The generic production system design process can also be viewed in terms of the specific types of systems involved, i.e., process, logistics support, and information. The remainder of this section briefly summarizes considerations associated with the design of each of these elements of the overall production system.

The design of the processing system involves: the selection of a hierarchy of processing systems to implement the modules (including plants, centers, lines, cells, stations, equipment, devices, and tooling), assignment of processes to the systems, estimation of resource utilization levels, and balancing of production systems.

The design of the logistics systems can be divided into two related problems: production material logistics and plant logistics. Production material logistics includes: determination of production material requirements (raw materials, components, packaging, carriers), estimation of consumption rates, determination of source selection strategies (make-or-buy analyses), lead times, and shipping (air/land/sea) methods for source materials.

Plant logistics concerns the systems which move and store materials within the facility. Plant logistics involves: determination of floor space and volumetric requirements for each process/machine/system, identification of production and tooling material storage requirements (i.e., loading docks, staging areas, centralized storage areas, line side storage), selection of storage systems (i.e., automated storage and retrieval systems, manual storage systems, production line buffers and feeders), specification of material flow through the facility (i.e., raw materials, components, work-in-process, and finished goods from the dock to lines through lines and back to dock), selection of material handling systems (e.g., hand truck, fork lift, conveyor, automated guided vehicles), determination of stock replenishment strategies, design of safety and environmental systems, development of physical plant layout in two and three dimensions, and evaluation of logistics system for further production capacity growth capabilities.

Production information systems may include: monitor and control systems, communications, display and user interface systems, database management systems and their
databases, data collection systems, production information systems, peripheral devices (e.g., printers, magnetic scanners, monitors, bar code readers, infrared tracking systems), production accounting and reporting, statistical process/quality control (SPC/SQC) systems, time and attendance recording, and preventive/corrective maintenance support systems. The information systems design activity includes: requirements specification, architecture development, process and information modeling, detailed design, interface specification, integration and test planning, and user documentation development.

The outputs of the production system design phase are detailed system specification documents. This phase may provide feedback to problem definition and process specification phases indicating changes which must occur as the result of design analyses. The next phase is the simulation modeling of the system which has been specified by production system design.

2.4 Simulation Modeling Of The System

Once a design, or partial design, for the production system is specified, it should be modeled and evaluated using simulation technology. The purpose of this phase is to better understand the dynamics of the proposed system and help ensure that it satisfies the constraints outlined in the problem definition phase. Inputs to this phase are derived from all of the previous phases. Pegden (1990), Askin (1993), and Carrie (1988) describe the simulation modeling process. Knepell (1993) describes the evaluation and validation of models.

The first step in developing a simulation model for the system is to define a problem statement and simulation objectives, i.e., what is expected to be learned from the simulation model. The types of alternative models to be considered and constructed need to be identified, e.g., discrete event simulation, material flow, system mechanics and kinematics, ergonomic, and/or manufacturing process. Appropriate simulation tools must be selected based on the types of models to be constructed. Next, system performance measures must be identified. Some examples of performance measures include: throughput, cycle time, work-in-process, machine downtime, and machine utilization.

Next, the system simulation model elements and their behaviors must be specified. Model elements used will depend on the types of simulations to be constructed. Elements of these models may include the attributes associated with: manufacturing resources, servers, queues and selection criteria, work pieces/loads/objects, arrival distributions, processes, system movements and material flows, timing distributions, failure and repair rates, etc. The information needed to derive the model elements will be drawn from problem definition, process specification, and system design data. The actual simulation models may then be constructed using the selected simulation tools.

Another critical activity in the modeling and evaluation phase is the development of test data for the simulation runs. This activity includes: identification of data sources, gathering of test data, formatting and loading the data, and determining the number of simulation runs required to produce significant results. Once the simulation has been constructed and the test data has been loaded, the models can be run and evaluated.

The simulations must be validated, i.e., it is necessary to determine whether results are believable based on experience, other data, etc. There are two aspects to this problem: 1) does the simulation program behave as expected, and 2) does the outcome reflect reality. If the results are not correct or creditable, either the simulation must be fixed, models modified, or the test data may need to be changed. Some examples of evaluations that may be performed on the results include: verification of the accuracy of model, analysis of errors and failures, bottlenecks, throughput, flow time, expected yields and quality, interference problems, collisions, etc.
After the results of the simulation are reviewed, it may be necessary to revise design specifications and the system models, process specifications, or even basic assumptions spelled out in the problem definition. Some of the results of simulation, e.g., timing data, may be fed forward into the engineering project management phase.

2.5 Engineering Project Management

Another parallel phase in production system engineering is the development of engineering project management data. Project management and budgeting is described in Kerzner (1984). These functions include: development of project plans, preparation of budgets, establishment of configuration management controls, and generation of reports. Principal inputs to this activity include: problem definition and system design specification data. Timing information may be drawn from simulation results.

Project planning involves the definition of the production system engineering project in terms of phases, tasks, resources, and timing data. Possible phases may include: feasibility study, planning, needs and requirements analysis, detailed design, acquisition and installation, testing, training, pilot and full production operation, and phase out. Critical milestones are identified as part of the phase definition activity.

Each major project phase is specified in terms of tasks and sub-tasks. Task precedence constraints and overlap options are identified. Required resources associated with each task are identified. Staff responsibilities are specified on each task. Resource balancing may be required. Timing information is also estimated for each task, including: expected or required start, end dates, estimated task durations and lead times. From this data, schedules may be generated and critical paths determined.

Cost factors and their analysis is an extremely important part of the system design and implementation process. Malstrom (1984) provides detailed guidance on the development of manufacturing cost estimates and budgets. Budget cost categories that may be considered include: project phase, planning, labor, tooling, capital equipment, projected maintenance, information and control system, operational, training, licensing and inspection, construction, installation, material (components, consumables), overhead (utilities, labor multipliers, area usage), and rental costs.

The budgeting process includes: gathering of cost data, entering data into spreadsheets or databases by budget categories, projecting estimates where data is unavailable, generating summaries by categories, and producing budget reports. Budgeting data is reviewed for significant deviations from targets and opportunities for savings are identified. Budget data is then used to generate feedback, if required, to the problem definition and production system design phases.

Another critical activity included in this phase is the configuration management of engineering data and project documents. Principles of configuration management are outlined in Daniels (1985). This activity includes: identification of key documents, definition of revision control-review-promotion policies and procedures, identification of organizational responsibilities, establishment of notification procedures for project staff, establishment of security policies and access control mechanisms, and the placement of documents and data under configuration management.

The final activity in the management area is generation and publication of reports that summarize the results of each of the other phases. Functions included in this activity include: outline development, document editing and assembly, layout and formatting, and printing. This activity draws input from all of the other functions in this phase and the other phase.
Once plans, budgets, configuration management policies, and reports are completed they need to be reviewed to ensure that they are realistic and meet the requirements established in the problem definition phase. If not, either the plans need to be changed or information must be fed back to problem definition and/or system design to re-scope the system.

3 INTEGRATED ENGINEERING TOOLS ENVIRONMENT

The process model for production system engineering is being implemented as an integrated tools environment through the collaborative efforts of NIST, academia, and industry. Black and Decker Corporation is collaborating on the production system engineering process and providing test data on production lines. Although a number of engineering tool vendors have provided software for integration into the environment, the final selections of all required software tools has not been completed.

The production engineering environment is being implemented on a high performance personal computer and Silicon Graphics™ engineering workstation. Commercial software tools used in the implementation of the engineering environment include: a business process re-engineering - flowcharting package, a plant layout system, a computer-aided design system, a manufacturing simulation system, a spreadsheet tool, a project management system, and a relational database management system. Other tools are under consideration for incorporation into the environment at a future time.

A virtual manufacturing system has been implemented using the simulation system to support the tool kit project. The virtual manufacturing facility currently provides simulation models to aid the engineer in the validation of engineering data. The virtual manufacturing facility currently consists of the following manufacturing areas (see Figure 3): tool room, shipping, receiving, heat treat, paint, manufacturing engineering administrative offices, and three machining areas.

The interoperability of the commercial engineering tools that are available today is extremely limited. As such, users must re-enter data as they move back and forth between different tools carrying out the engineering process. Project collaborators are working with NIST to address this problem. Collaborators will: define generic information models for production system engineering data, specify interfaces for integrating tools, develop prototype integrated environments and shared databases, and implement test case production system engineering projects. Examples of the types of shared data under consideration by the collaborators for the common database includes: production requirements, product specifications, process specifications (diagrams, flowcharts, plans, routings, operation sheets, programs), equipment specifications, budget spreadsheets, project plans, simulation models and model elements, setup illustrations, plant layouts, information models, interface specifications, system descriptions, estimated yield data, process capabilities, and quality data.

A long term objective of the project is to improve the productivity of users by creating an integrated environment where changes to data and decisions automatically percolate through the various tools contained within system. Project results will provide a basis for defining interface standards that will facilitate the integration and interoperability of commercial tools in the future.
Figure 3  Virtual manufacturing system implemented at the National Institute of Standards and Technology.

The author wishes to acknowledge contributions to this paper from Mike Iuliano, Swee Leong, and Dr. Albert Jones. Work described in this paper was sponsored by the U.S. Navy Manufacturing Technology Program and the NIST Systems Integration for Manufacturing Applications (SIMA) Program. No approval or endorsement of any commercial product by the National Institute of Standards and Technology is intended or implied. The work described was funded by the United States Government and is not subject to copyright.

4 REFERENCES

Manufacturing engineering software integration


5 BIOGRAPHY

Charles McLean is Leader of Manufacturing Systems Engineering Group in the Manufacturing Systems Integration Division at the U.S. National Institute of Standards and Technology. He has managed research programs in manufacturing automation at NIST since 1982. He has served as System Architect for the Automated Manufacturing Research Facility (AMRF) and Program Manager for the National PDES Testbed. His group is conducting research in engineering tool integration, manufacturing data validation, production systems engineering, manufacturing simulation, and production scheduling. He holds a Master of Science in Information Engineering from University of Illinois, Chicago, IL, and a Bachelor of Arts degree from Cornell University, Ithaca, NY.
Trends in Manufacturing Control

P.J.M. van Bommel
Philips CFT Production Systems
PO Box 218
5600 MD Eindhoven
The Netherlands

Abstract
Within Philips the Centre for Manufacturing Technology (CFT) is positioned as a R&D organization, focused on the manufacturing function in the company. The Production Systems department focuses on the specification and design of new production systems and the improvement of existing operations.

The following trends strongly affect manufacturing:
• From manufacturing to mento-facturing.
• Making use of the hands, but above all the brain power of the workers. This has large implications for the factory organization (example: mini-companies). Therefore it affects also the information flows on the shop floor. When taking into account the trend "from make to buy" it is clear that hierarchical architectures, like the CAM Reference Model (fig. 5), are not suitable anymore. It is replaced by models fitting better to applications available on the market (the Application Reference Model).
• From mass production to mass customization.
• Groups of customers desire personalized products at mass production prices. This implicates multiple manufacturing concepts in the same production organization. The logistics aspects are becoming closely entangled with production aspects.
• Integration of production in the overall business
• Leading to concurrent engineering, extensive work flow control and closer relations between product design and production system design.

Keywords
Shop Floor Control, Architecture, Human aspects, System development approach
1 MANUFACTURING EXCELLENCE

Recently, the manufacturing function within Philips Electronics is receiving a lot of attention. It is recognized that manufacturing is of vital importance for Philips. In order to survive Philips needs to reach world class manufacturing standards [1]. The manufacturing function is reinforced by benchmarking plants with others, both internal Philips and external. This benchmarking is done by means of the so-called Manufacturing Excellence program [2], stating a set of aspects and goals to achieve.

The aspects covered are:
- Organization and culture
- Cycle times
- Quality / customer satisfaction
- Financial measures
- Physical plant
- Product creation process
- Supplier-base management

When scanning through the benchmark list of aspects at a detailed level, emphasizing information and control aspects, we recognize the following trends/issuses:
- Empowerment of employees and deployment
- Total employee involvement
- Awareness for customer satisfaction
- Customer involvement
- Integrated information and production systems
- Concurrent product and process engineering
- Order information integrated with manufacturing system
- Supply synchronized with production, without contributing to excess inventory or WIP
- Planning resolution and frequency Â 1 day
- Set-up / change-over times of minutes
- Continuous flow production
- Integrated automation
- CIM with enterprise-wide information interface
- Only value-added automation
- Integrated information system; Production information shared with all employees and linked to company goals
- Maintenance synchronized with production

Based on the observations mentioned above some trends are derived that influence production information systems.

2 ROUGH CHARACTERIZATION OF PHILIPS MANUFACTURING

Philips manufacturing has the characteristics of discrete manufacturing: Relatively high volumes of individual products are mass produced in assembly type of plants. Philips plants
are located world wide. Often the product life cycle is shorter that that of the production means. This requires flexible production systems. As humans are the most flexible elements in manufacturing, human involvement in production is essential. As the shop floor is continuously adapting, it is important to keep changes local in order to keep them manageable. Therefore often a decentralized control architecture is applied. Figure 1 shows some general trends affecting Philips products and production.

![Figure 1: Issues and trends in discrete manufacturing](image)

![Figure 2: Typical control loops in discrete manufacturing](image)
Figure 2 shows the typical control loops in discrete manufacturing environments. Essential is:

- The interaction between the improvement cycles, in terms of production and process improvement and development of both product and process.
- The operational order structure of the factory.

Automated support of discrete manufacturing is relatively new. The first area for which software systems for production control were available is continuous production (for example refineries). These type of plants are controlled from a central control room, where all sensors are monitored and actuators, like valves, etc. are controlled. Factory automation concepts started with the central control paradigm [3]. From this centralized form of control the decentralized concepts are derived, especially in the discrete manufacturing environment where flexibility is paramount. See figure 3.

![Diagram of control structures](https://via.placeholder.com/150)

**Figure 3:** The trend from centralized to (ultimately) heterarchical control structures in manufacturing

3 FROM “MANU”FACTURING TO “MENTO”FACTURING

The first trend recognized on the factory floor is the idea to make full use of the brains of the “blue collar” workers. In former manufacturing paradigms workers were performing short cyclic tasks. They were not supposed to think. People were seen as robots, with the disadvantage that they make ‘unpredictable’ mistakes. At present workers are regarded as the essential capital of a manufacturing enterprise [4, 7]. Not only their physical skills are useful on the factory floor, but also their mental skills.

As is illustrated in figure 4 [5], the quality that is perceived by the customers of a manufacturing enterprise is largely influenced by the people in that enterprise. The people are organized to fulfill the basic processes:

- Product and process development
• Manufacturing
• Customer support and after sales

Essential in this view is the communication between the people. This is illustrated by the fact that words like People, Teamwork and Organization are directly linked. It is of vital importance that knowledge is shared between all people working in a manufacturing organization.

Figure 4: The new manufacturing wheel, emphasizing the central role of people in the manufacturing industry

Focusing on operators on the factory floor, the following trends are recognized:
• The number of operators is decreasing
• The number of machines per operator is increasing
• The functionality and flexibility of machines and therefore their operation complexity is increasing.
• Often the operator is regarded as the 'owner' of his machines, suggesting he has the final word regarding everything that has to do with his machines. The operator therefore has to acquire the skills of a service engineer, quality engineer, etc.
As a reaction to these trends especially the user interface of the (increasingly “intelligent”) machines is adapted. Knobs and dials oriented and character oriented user interfaces are replaced by graphically oriented user interfaces. At the same time a trend towards multimedia is visible. In this context multimedia is defined as the integrated application of video (graphics, photo’s and movies) and audio (Signals and spoken text).

An implication of using the mental skills of the workers is the learning process these workers go through. One of the ways to enable this learning process is to delegate responsibilities and authorities as far down in the organization hierarchy as is possible. This is done by establishing autonomous work groups, also called “mini-companies”. Such a mini-company is an operationally oriented unit covering a section of the production chain, aimed at:

• Running the operations as an independent team
• Local improvement and optimization of production

Some essential characteristics are team spirit, a quality drive and use of the practical know how, available on the factory floor.

With respect to the information supply of these mini-companies the following remarks can be made:

• Presentation aspects are important, as performance data are presented to the mini-company members and to others via publication boards (glass wall management)
• The information requirement is continuously changing, due to the learning curve of the mini-company
• Information must be supplied at the right level of abstraction
• Information must be made available locally
• Information on the information must be available (meta information, stating the meaning of the data presented)

4 FROM MAKE TO BUY

Up to now, in discrete manufacturing, production control systems architecture is predominantly hierarchical. In the continuous process industry centralized architectures are used, as is illustrated in figure 2. In cooperation with Digital Equipment Corporation, Philips has developed the CAM Reference Model [6]. This is also an hierarchical architecture as is shown in figure 5.

One of the major disadvantages of the CAM Reference Model is the strictly logical separation of functions between the levels. The CAM Reference Model is used as a blue print of the manufacturing control system architecture during requirements engineering and analysis phase. Never the less the structure of the ultimately implemented control system is often disjunct from the CAM Reference Model. This is due to the fact that system components, available on the market, are not designed according to the CAM Reference Model. The CAM Reference Model is a logical control model, while the ultimate implementation of the control system is according to a IT implementation architecture.

As Philips has no competitive edge in the development of manufacturing control systems it is wise to use solutions available in the market as much as possible. The CAM Reference Model still is an adequate tool to guide the system analysis phase of a shop floor control system development projects, but is far from available components for these systems.
Therefore the CAM Reference Model is replaced by the “Application Reference Model”. This model starts with roughly the functionality offered by software packages available in the market. Of course the functionality offered by packages often overlaps that of other packages. But, as a single package solution is not feasible for all types of manufacturing within Philips, the concept of multi-vendor application packages appears sensible. The Application Reference Model is shown in figure 6.

The essential elements of the Application Reference Model are common application packages available in the market. Examples are:

- Enterprise Resource Planning: Triton (Baan/4), SAP R/3, MFG/PRO
- Engineering Data Management: Sherpa, HP Workmanager
- Scheduling: MOOPI, FI-2, Pacemaker, Visual Manufacturing
- Maintenance Management: SAP maintenance module, MAXIMO
- Manufacturing Execution 1: Intrack, Workstream, Factoryworks, MESA
- Engineering Data Analysis: IDS
- Supervisory (Cell) Control (SCADA): Intouch, FIX-DMACS, Factorylink, Wizcon

None of the vendors of these packages can offer a standard approach to arrive at an integrated system covering the complete scope of manufacturing, although they tend to claim more than they actually offer. A trend is seen towards vendors embarking in partnerships in order to cover the whole set of requirements with their integrated set of solutions. An application reference model which describes the complete scope of applications and which relates the application packages to each other therefore is necessary.

When focusing on the system development aspects associated with the use of common application packages available in the market it is recognized that the conventional waterfall type of system development (Figure 7) is not feasible any more. It should be replaced by an incremental approach as shown in the same figure 7. The latter approach uses the Application

---

1 In literature, maintenance management and scheduling packages are often considered to be part of manufacturing execution.
reference Model to select application packages covering the key requirements with as little overlap between the functionality of the packages as is possible (i.e. an application framework). When application packages have been selected, the first step is to customize these packages according to the business processes in the manufacturing organization. In terms of these business processes the application functions should be mapped. Selection and customization of a package is the balancing of the user requirements with the restrictions posed by the packages.

![Diagram of the Application Reference Model](image)

**Figure 6: The Application Reference Model, recently developed by Philips**

As a consequence of the system development approach and the dynamic nature of the behaviour of the shop floor (a characteristic of discrete manufacturing), it must be possible to implement or change functionality quite rapidly. This implicates that application packages must be:

- Configurable, being adaptable to the information needs
- Scaleable, being extendable for the size of the organization
- Pluggable, being exchangeable by other more suitable application
- Inter-operable, being able to work together to exchange information using the functionality of several applications.

Therefore, in addition to an application reference model, also a technical architecture is required as part of a CIM framework. A powerful link for shop floor control is an information bus. An information bus provides the interfaces for communication between applications, hiding these interfaces from platform details, like hardware, network and
databases. Thus we arrive at a preferred technical architecture of shop floor control systems as is shown in figure 8.

![Diagram showing the conventional and incremental system development approaches.](image)

**Figure 7: The conventional and the incremental system development approaches**

![Diagram showing the preferred technical architecture of shop floor control systems.](image)

**Figure 8: The preferred technical architecture of shop floor control systems**

This architecture is based on a mechanism of publishing of, and subscribing to information available in the complete infrastructure. In this way real-time information on order, material, quality, equipment, etc. can be made available and used throughout the manufacturing environment. This event-based approach is indispensable in a shop floor control environment.
and should be regarded as an addition to the demand-based approach of conventional client-server architectures.

5 FROM MASS PRODUCTION TO MASS CUSTOMIZATION

One of the present challenges, especially in the consumer electronics industry, is the changing paradigm from mass production to mass customization. Personalized products are required, for the price of mass produced ones. This leads to a paradox between the market requirements and the prerequisites for efficient production. This illustrated in figure 9.

<table>
<thead>
<tr>
<th>market requirement</th>
<th>efficient production</th>
</tr>
</thead>
<tbody>
<tr>
<td>order throughput time</td>
<td></td>
</tr>
<tr>
<td>&lt; 24 h</td>
<td>&gt; 1 week</td>
</tr>
<tr>
<td>order size</td>
<td></td>
</tr>
<tr>
<td>&lt; 100</td>
<td>&gt; 2000 (1 shift)</td>
</tr>
<tr>
<td>product diversity</td>
<td></td>
</tr>
<tr>
<td>&gt; 5000</td>
<td>&lt; 100 (1 / shift / line)</td>
</tr>
<tr>
<td>product life time</td>
<td></td>
</tr>
<tr>
<td>&lt; 6 months</td>
<td>&gt; 2 years</td>
</tr>
</tbody>
</table>

Figure 9: The market - manufacturing paradox

The market essentially asks for maximum flexibility of the manufacturing. The manufacturing system itself opposes flexibility as it is required to operate at minimum cost. A way to satisfy these contradicting requirements is the definition of a decoupling point between:

- A mass production factory, with a limited product diversity and a high degree of mechanization / automation
- A highly flexible factory, being able to handle high diversity. Very often this is a flexible manual final assembly operation

Depending on the nature of the product it even is possible to distinguish a third factory type:

- A combination of both types mentioned above, able to cope with moderate product diversity. See figure 10.

In figure 10 the triangles are decoupling points between types of production. The first two, PCB assembly (+ testing) and Sub-assembly (+ testing), are planning driven, while the third one, Configuration, is the demand driven final assembly operation.
One of the essentials of this type of manufacturing is that the architecture of the product and production system influence each other directly. This implicates that the manufacturing system must be designed in conjunction with the product (family). The systems used on the shop floor, especially the control systems, must be highly flexible. The incremental system development approach, based on the application reference model fits closely to the demands of such manufacturing systems.

6 CONCLUSIONS

Some conclusions to be drawn are:

• The drive for Manufacturing Excellence within Philips is essential to achieve world class manufacturing

• The role of people in the manufacturing process is recognized and emphasized. The control and information systems must fulfill the information requirements of the people on the factory floor as they learn.

• As the role of people on the shop floor is emphasized, the interfacing between humans, machines and control systems is becoming more important. New technology evolves, ultimately leading to multi-media type of human interfaces.
• Information and control systems play an important role in manufacturing and increasingly will do so.
• The incremental system development approach, based on common application packages available in the market, using the application reference model as guideline, offers excellent opportunities to cope with the requirements of modern manufacturing:
  • Emphasis on the role of people on the shop floor
  • Trend to mass customization
  • Continuous reduction of business cycles

7 REFERENCES

[4] Philips Electronics; The Philips values; 1995
PART TWO

Modelling of Products, Processes and Systems
Integration of product life cycle views on the basis of a shared conceptual information model

Prof. Dr.-Ing. Reiner Anderl  
Dipl.-Inform. Anne Wasmer  
FG Datenverarbeitung in der Konstruktion (DiK)  
FB Maschinenbau, TH Darmstadt  
D-64287 Darmstadt, Petersenstraße 30  
Telephone: 06151/16-6001, Fax: 06151/16-6854  
email: anderl@dik.maschinenbau.th-darmstadt.de  
wasmer@dik.maschinenbau.th-darmstadt.de

Abstract
The paper presents the development and modelling approach of ISO 10303 ('Product Data Representation and Exchange') with a focus on the mapping approach currently taken. As result of the mapping process, the so-called application interpreted model (AIM) is created that is dedicated to be used as a basis for implementations. The paper describes an approach how the AIM schema which is the target of the mapping process and which forms the basis for integration can be automatically derived and how this approach then can be combined with existing mapping languages suggested in the STEP arena with the goal to integrate application specific life cycle views in an enterprise.

Keywords
application protocol, unit of functionality, mapping, mapping table, mapping language, view

1 INTRODUCTION

Due to the increasing need for computer systems support during all phases of the product life cycle, a lot of computer aided systems have been developed and been used during the last years. They are usually specialized to support a specific application and are designed based on their own individual information model representing their particular application domain view on the product and the product model respectively. As a consequence, these application systems build
islands of automation and as such don’t allow for the joint use of the product data described by those information models. The result is on one hand a wealth of information models tailored to meet specific application requirements and overlap in the semantic contents they are representing, and on the other hand the redundancy of product data stored in the different application systems which makes it difficult to maintain the product data and keep them consistent during the product life cycle. Nowadays, new working methods and techniques like total quality management, concurrent and simultaneous engineering, together with an increasing need for co-operation between enterprises already during the development and design process require continuous process chains with an integrated information technology support throughout the whole product life cycle. Product data technology plays a key role in this context. The development of ISO 10303 (STEP) is an example for the product data technology approach to create an integrated information model representing all information necessary to describe a product throughout its whole life cycle. As the purpose of this model is to support individual applications or process chains, the concept of the so-called application protocols has been introduced. An application protocol (AP) in ISO 10303 defines all information needed to support a particular process chain or application. The definition of an ‘application’ to be ‘a group of one or more processes creating or using product data’ (ISO 10303-1, 1994) has led to the definition of application protocols with a significant difference in the size of their scope. While some APs define their scope as being the context for one computer application independently from a certain type of product, e.g. AP202 ‘Associative Draughting’ (ISO 10303-202, 1996), others deal with complete process chains involving a lot of different computer applications, e.g. AP214 ‘Core data for automotive mechanical design processes’ (ISO TC184, 1995-2), that covers the development of automotive products including the definition of requirements, styling, design, evaluation, production planning of the automotive parts as well as design and manufacturing of the tools used to produce them. Depending on the scopes of the existing APs, computer applications used during the life cycle of a product can be integrated on basis of one single AP or multiple APs can be combined to cover the information requirements of the computer applications to be integrated. In the following, it is shown how the three layer architecture of APs can be combined with existing mapping languages suggested in the STEP arena and be used as the basis to integrate specific views of computer applications throughout a process chain.

2 ARCHITECTURE OF ISO 10303 APPLICATION PROTOCOLS

The objective of ISO 10303 is to provide a neutral mechanism capable of describing product data throughout the whole life cycle of a product, independent of any particular application system. Although ISO 10303 has been designed to support the product data exchange, its nature makes it also suitable as a basis for implementing shared product databases and archiving. While product data exchange is defined as the transfer of product data between a pair of applications that operate upon their own copies of the product data, product data sharing is characterized by more than one application logically operating on a single copy of the same product data (ISO TC184, 1996-2).

ISO 10303 is organized in different series including description methods (10 series), containing e.g. the formal information modelling language EXPRESS (ISO 10303-11), implementation methods (20 series), methods for conformance testing (30 series), and the product data specifications that are divided in three categories: integrated resources (40 and 100
The integrated resources are a coherent set of modularly structured schemas specified in EXPRESS. They can be seen as an integrated conceptual reference model for product data. They are defined independently of a specific application context in which product data might be used. The extension of the Integrated Resource models is done in a synergistic process following the integration methodology described in (ISO TC184, 1996-2). Its purpose is to maintain the Integrated Resources as a single, self-consistent model (consisting of different schemas) free of redundancy.

Application protocols are defined for a specific industrial application domain or application context specifying the functional aspect of the information, i.e. the collection of processes that is supported by the application protocol, the types of product or industry sector, the life cycle stage and discipline. An application protocol is based on a three layer architecture and consists of three types of models described in the following.

- The application activity model (AAM) specifies the enterprise activities ('what is done?' versus 'how is it done?'), that are constituting the application context, and the information flows between them in IDEF0.

**Figure 1** Components of an Application Protocol.
• The **application reference model** (ARM) details the information flows identified in the AAM. It specifies the information requirements of the application protocol, i.e. for the defined application context, in the terminology of the application domain and structured in the way the domain experts think of it. It is graphically represented as an information model in EXPRESS-G, IDEFIX or NIAM together with a textual definition of the identified so-called application objects, i.e. entities and their attributes (application objects), and their relationships (application assertions).

• The **application interpreted model** (AIM) is based upon the semantic content of the ARM. The AIM is specified in EXPRESS and used as a basis for implementations of product data exchange and also forms the conceptual basis for other implementations like shared databases. It consists of Integrated Resource constructs and specializations of them that semantically correspond to the application domain's information requirements (described in the ARM) and of constraints to restrict the population of the AIM according to the means of the particular application domain. These correspondences are identified during the so-called interpretation process (mapping) and documented in the mapping tables.

When two or more application protocols have an overlap in their information requirements, they have to use the same interpretation, i.e. the same set of Integrated Resource constructs and their specializations to fulfil these requirements in the AIM. In this case, the set of common AIM constructs is specified in a separate EXPRESS schema, the **application interpreted construct** (AIC). All application protocols which have been identified as sharing a common information requirement use the same application interpreted construct in its entirety.

### 3 METHODOLOGY OF ISO 10303 APPLICATION PROTOCOL DEVELOPMENT

After defining the textual definition of its scope and identifying the application context, the application protocol is developed in phases that comprise AAM, ARM, mapping table, and AIM development. The phases of AP development are described in detail in (ISO TC184, 1996-2). In the following, the main focus is on those concepts that are considered to be essential from an application integration perspective. To illustrate the concepts, examples from AP214 (ISO TC184, 1995-2) are provided.

To handle the complexity and to provide a mechanism for modularizing the AP on the requirements definition level, ARMs are structured in so-called Units of Functionality (UoF). A unit of functionality is a collection of entities, their attributes, and relationships, that conveys one or more well-defined concepts within the context of an AP's ARM. The ARM constructs grouped into a UoF build a logical unit that usually supports an application function or process. Modularizing the ARM into units of functionality facilitates on one hand the development of the ARM itself as a clearly structured model without redundancy, on the other hand it helps detecting information overlap between different application protocols and as such facilitates their integration. Depending on the application context, i.e. the collection of processes that is supported by the application protocol, an ARM can contain one or more units of functionality. All units of functionality are related to the processes defined in the AAM, that require each functionality. Figure 2 shows the relationship between data classes identified as information flows within the AAM and the UoFs derived from these data classes.
In the case of AP214, the ARM is divided into 35 UoFs. These UoFs represent modules that contain e.g. basic product management data, work management data, process plans, geometric models such as brep or surface models, surface conditions, geometric tolerances and different types of form features.

Once the overall structure of the ARM into UoFs has been fixed, the ARM is developed with the input of the application domain experts and based on their terminology detailing the information flows of the AAM identified to be in scope. The development process has to ensure integrity and self-consistency of the ARM that has to be an integrated conceptual model for the considered application domain. An overview of current methodologies for the development and integration of such models can be found in (Polly, 1996). To ensure consistency not only within one, but among multiple application protocols, the reuse of the common integrated resources by the mechanisms integration and interpretation have been introduced. The
Part Two  Modelling of Products, Processes and Systems

fundamental principle of the data integration aspect is the common use of product data constructs provided by the Integrated Resources across the AIMS of all application protocols. During the interpretation process, the information requirements described in the ARM (application objects and assertions) are mapped to suitable constructs of the Integrated Resources, where necessary refinements and/or restrictions in form of new subtypes and constraints (local or global rules) are identified. It is part of the integration task to identify potential overlaps with other application protocols, that manifest themselves in the form of overlap in the AAMs, similar UoFs, and/or correspondences between the ARMs, and to ensure that same information requirements result in the same selection and specialization of integrated resource constructs. In case the overlaps result in new subtypes, AICs containing these new subtypes have to be defined and incorporated into the AIM.

The result of the interpretation process is documented in two components of the application protocol, the mapping table and the AIM. The mapping table provides the correspondences between the application specific information requirements described in the ARM and the constructs of the AIM that fulfil those information requirements and thus ensure traceability of the product information to be exchanged or shared on basis of the AIM back to the information requirements specified for the specific application domain in the ARM. A mapping table contains one table per UoF that is organized into five columns. The first column contains a separate row for all application elements, i.e. application objects (i.e. ARM entities and attributes) and assertions (i.e. relationships between entities) of the UoF, in alphabetical order of the entities. Attributes and assertions directly follow the entities for which they are defined. The second column contains the AIM element(s) to which the application element maps with the indication of the source schema in which the AIM element has been originally defined. Possible sources can be the integrated resource Parts, AICs or the application protocol itself. Column 4 of the mapping table contains numerical references to the global rules that constrain the valid instantiation of the given mapping in the AIM. The numerical references correspond to a numbered list of all global rules defined in the AIM.

Because of their different structure and terminology and the potential use of different modelling languages, there is often not a direct correspondence between ARM and AIM constructs. In case the particular ARM requirement is specified on a lower level of detail than the integrated resources, the mapping of an application element can require a combination of AIM elements (AND situation) or give alternative mappings (OR situation). In addition, it might be necessary to express the particular application element by a related set of AIM constructs. How the AIM constructs have to be related to fulfil a particular information requirement is specified in column 5, the so-called reference path column. A reference path syntax has been developed to document the AIM structure necessary to fulfil the information requirement described by the particular application element.

In case an Application object maps to an AP specific subtype, the Reference path column contains the reference path from the subtype to its supertype from the Integrated Resources. In case the application element is a simple attribute that maps to an attribute of another AIM entity than the Application object maps to, the Reference Path column shows the reference path from the AIM element of the Application object to the AIM element of the simple attribute. In case the application element is an assertion, i.e. a relationship between to ARM entities that map to different AIM constructs, the AIM element column contains the word 'Path' and the reference path column documents the AIM constructs and their relationships necessary to establish the relationship on the AIM level, i.e. it contains the reference path starting at the AIM element the "From" Application object maps to and ending at the AIM element which the "To" Application
Integration of product life cycle views

object maps to. Mapping rules can be used to specify that attribute values are restricted or certain subtypes are required only for a certain reference path.

The symbols used within a reference path definition, are shown below.

[ ] : Multiple AIM elements or sections of the reference path are required to satisfy an Information requirement (AND situation).

() : Multiple AIM elements or reference path sections are (exclusive) alternatives within the mapping to satisfy an information requirement (OR situation).

{} : Enclosed section constrains the reference path to satisfy an information requirement (Mapping Rule).

-> : Attribute references the entity or select type given in the following row.

<- : Entity or select type is referenced by the attribute in the following row.

[i] : Attribute is aggregation of which a single member is given in the following row.

=> : Entity is a supertype of the entity given in the following row.

<= : Entity is a subtype of the entity given in the following row.

\ : Line continuation character.

1) : Footnotes are used to provide additional information.

newline : Entity has attribute documented in the following row or attribute is attribute of entity documented in the following row.

A more detailed description of the mapping table documentation can be found in (ISO TC184, 1996-1) and (Anderl, 1995). As an example, a simplified extract of the ARM is shown in figure 3. The corresponding mapping table for this ARM extract is shown in figure 4.

Figure 3  Simplified extract from the mapping table of AP214.

Alongside the development of the mapping table, a so-called shortform of the AIM is developed. It contains references to the integrated resource constructs and to the AIC schemas to which a mapping exists (in form of the 'USE FROM' construct defined in EXPRESS), as well as the (textual and EXPRESS) definitions of AP specific types and entities and the global rules to restrict the valid sets of instances for the application domain. With the integrated resource and AIC schemas as input, this AIM shortform can be expanded to a self-contained longform according to the interface rules defined in (ISO 10303-11, 1994). This step might bring in entities, that are not specified in the AIM element or reference path column, but referenced by other entities.
### Mapping table form_feature_with_explicit_shape UoF (FF3)

<table>
<thead>
<tr>
<th>Applic. element</th>
<th>AIM element</th>
<th>Source</th>
<th>Rules</th>
<th>Reference path</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOLE_BOTTOM_CONDITION</td>
<td>hole_bottom</td>
<td>214</td>
<td></td>
<td>hole_bottom &lt;= shape_aspect</td>
</tr>
<tr>
<td>bottom_type</td>
<td>shape_aspect.name</td>
<td>41</td>
<td></td>
<td>hole_bottom &lt;= shape_aspect</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>shape_aspect.name = 'radiused'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(shape_aspect.name = 'flat_with_radius')</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(shape_aspect.name = 'flat')</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(shape_aspect.name = 'pointed')</td>
</tr>
<tr>
<td>ROUND_HOLE</td>
<td>round_hole</td>
<td>214</td>
<td></td>
<td>round_hole &lt;= machining_feature &lt;= shape_aspect</td>
</tr>
<tr>
<td>maximum_depth</td>
<td>measure_representation_item</td>
<td>45</td>
<td></td>
<td>round_hole &lt;= machining_feature &lt;= shape_aspect</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>shape_aspect = shape_definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>characterized_definition = shape_definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>characterized_definition &lt;= property_definition.definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>property_definition &lt;= property_definition_representation.definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>property_definition_representation_representation &lt;= property_definition_representation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>representation.used_representation =&gt; representation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>{representation =&gt; shape_representation}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>representation.items[i] =&gt; representation_item =&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>{representation_item.name = 'maximum_depth'}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>measure_representation_item</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>{measure_representation_item &lt;= measure_with_unit =&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>length.measure_with_unit =&gt;</td>
</tr>
<tr>
<td>round_hole to hole_bottom_condition</td>
<td>PATH</td>
<td></td>
<td></td>
<td>round_hole &lt;= machining_feature &lt;= shape_aspect</td>
</tr>
<tr>
<td>(as hole_bottom_condition)</td>
<td></td>
<td></td>
<td></td>
<td>shape_aspect &lt;= shape_aspect_relationship.related_shape_aspect</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>{shape_aspect_relationship =&gt; feature_component_relationship}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>shape_aspect_relationship.related_shape_aspect =&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>shape_aspect =&gt; hole_bottom</td>
</tr>
</tbody>
</table>

**Figure 4** Simplified extract from the mapping table of AP214.

Figure 5 shows an EXPRESS-G diagram of the AIM derived from the example given in figures 3 and 4. It can easily be seen, that the result of the mapping from the ARM to the integrated resources that are defined independently from a specific application context, is a more complex
structure that is more difficult to understand than the original structure specifically defined for the application context.

**Figure 5** AIM subset derived from the figure 4 example (simple attributes not shown).

**4 GENERATION OF AIM AND AIM SUBSETS FOR CONFORMANCE CLASSES**

Although the mapping tables as currently documented in the application protocols are only described in a semiformal way, they allow for the automatic derivation of the information structure and part of the constraints of the AIM, e.g. which AIM constructs are existence dependent from other constructs. For this purpose, a tool has been developed, that is based on an own syntax format describing the mapping table. This tool has already been used to generate the EXPRESS shortform for the complete AIM of AP214 and also to create specific AIM subsets dedicated to the needs of particular computer applications used within the process chain supported by the AP. The subset generation is done on the basis of conformance class definitions which are based on the UoF structure of the ARM.

A conformance class is a definition of a subset of the AIM that may be used as the basis for implementation. In order to meet the needs of different computer applications used within a given application domain, two or more conformance classes may be defined for an application protocol, where if no conformance class is specified, a conforming implementation has to implement the complete AIM. As the definition of an implementation specific subset should be done from the application’s point of view, that is described in the ARM, the AP214 approach is to specify the conformance classes on the ARM level, i.e. as combinations of UoFs. To derive the corresponding AIM subsets, that have to serve as the basis of the actual implementations,
the correct alternatives have to be extracted from the information contained in the mapping
tables. For this purpose, the mapping table had to be extended by some additional information
e.g. capturing the dependencies and/or correspondences between the implementation specific
view, i.e. the conformance class definition based on the UoF structure, and the alternatives in
the mapping documented as OR cases in the mapping table. Figure 6 shows the input and
output for the AIM and conformance class generation.

Figure 6 Input and output for the generation of implementation specific conformance classes.

Although the current mapping table definition serves with some extensions as basis for the
generation of the AIM subsets for the conformance classes, which provide the normative
specification for the product data to be exchanged between computer applications that support
the particular conformance class and also the conceptual schema for data sharing
implementations, it still has some lacks e.g. the fact that the current mapping syntax is not
specified in an unambiguous and formal way, that it cannot be exactly described how an
instantiation of the AIM for a given ARM instantiation has to look like, and that a bidirectional
mapping, i.e. from ARM to AIM and vice versa, cannot be specified.

In parallel to the current efforts to improve the mapping table syntax used, there have been
ongoing efforts in and outside of the STEP arena to define mapping languages, that formally
define how instances of a mapping target schema can be generated out of an instantiated source
schema (unidirectional mapping) and eventually vice versa (bidirectional mapping). They are
based on different approaches and therefore suited to perform in different areas. Examples of
such mapping languages are EXPRESS-V (ISO TC184, 1995-3), EXPRESS-M (ISO TC184,
1995-1), and EXPRESS-X (ISO TC184, 1996-3). As an example, EXPRESS-V is mentioned here that seems to be suitable to compensate some lacks of the current mapping table syntax definition. A possible usage scenario for the integration of computer applications based on a single application protocol is described together with a proposal how the scenario can be realized using the information contained in the mapping tables.

5 GENERATION OF APPLICATION SPECIFIC VIEWS

EXPRESS-V has been developed to accommodate simplified views of a product model that omit unnecessary details of the model, that are not needed by particular computer applications. Using such a view is conceptually easier to understand and process within the application system. As the applications also need the possibility to update the data they access through their view, the mapping proposed is bidirectional.

Within AP214, the complete ARM represents an integrated schema to describe the information requirements of all applications supported by the AP on a conceptual level. An application specific view can be seen as one conformance class definition on the ARM level, i.e. as combination of those UoFs that are supported by this particular application. As described above, an implementation specific AIM subset corresponding to a set of conformance classes supporting a given set of applications to be integrated can be derived from the mapping table. It can be used as the integrated schema for a data sharing implementation for this set of applications. To generate the application specific views on the implemented schema, a mapping is necessary from the implementation back to the application requirements. Such a mapping can be specified using EXPRESS-V. The mapping table documented in the AP seems to be a good starting point for this task. Although it does not capture the complete instantiation relationships between ARM and AIM, it describes their semantic correspondences in an easy understandable way and ensures that an unidirectional mapping exists for all ARM constructs. Current work deals with extending the mapping table specification to capture more semantics to document the relationships between the original application specific requirements and their corresponding AIM constructs in a more precise way in order to derive a 'backwards' mapping from the AIM, i.e. the schema used for implementation, back to the application specific requirements specified in the ARM. Based on these extensions, EXPRESS-V schemata can at least be partially derived, each of them specifying the mapping between the AIM schema that acts as basis for the data sharing implementation, and a conformance class specified in the terminology and from the viewpoint of a specific application as combination of UoFs on the ARM level. From the completed EXPRESS-V specifications software can be derived to perform the mapping at runtime.

6 CONCLUSIONS

The integration of product life cycle views on the basis of an integrated product model specification becomes more and more important. Particularly for the support of product development processes, views dedicated to support the various types of applications used during the different life cycle phases are strongly required. The approach of integrating product life cycle views on the basis of a shared conceptual information model takes into account the data specification levels of ARM and AIM introduced by the current ISO 10303 architecture.
Although the mapping table notation as currently used to document the ARM to AIM mapping has several lacks, it is relatively easy to understand and it provides a systematic approach and structured way to document the mapping ensuring that for each construct of the source model, i.e. the ARM, a mapping specification exists to the target model, i.e. the AIM. This AIM does not exist at the beginning of the mapping process, but is created during this process ‘on the fly’. Therefore, the mapping table is considered to be a suitable starting point for the mapping process, that has to be extended by further capabilities such as capturing instantiation correspondences. The extended mapping table can then be used to generate (at least partially) mapping schemas in a formal mapping language like EXPRESS-V or EXPRESS-M, depending on the purpose of the implementation to be supported. Out of this formal specification, software can be derived to perform the mapping. Current work in this area include the specification of extensions to the mapping table syntax in this direction and the development of tools that derive such formal specification based on the mapping table notation.

7 REFERENCES


8 BIOGRAPHY

Prof. Dr.-Ing. Reiner Anderl received his PhD in 1984 and was working as chief engineer at the Institut für Rechneranwendung in Planung und Konstruktion (RPK) at the University of Karlsruhe. In 1991 he got the appointment to a university lectureship in the area of CAD/CAM and product data technology. Since April 1993 he is professor for computer integrated design at the Fachgebiet Datenverarbeitung in der Konstruktion DiK at the University of Darmstadt.

Anne Wasmer received her degree in computer science at the Technical University of Karlsruhe where she worked as research assistant at the ‘Institut für Rechneranwendung in Planung und Konstruktion’ (RPK) since 1992. Since April 1993, she has been working as a research assistant at the ‘Fachgebiet Datenverarbeitung in der Konstruktion’.
CIMOSA process model for enterprise modelling

Kurt Kosanke(1), François B. Vernadat(2), M. Zelm(1)
(1) CIMOSA Association e.V., Stockholmerstr. 7, D-71034 Böblingen, Germany
(2) Université de Metz, LGIPM, Ile du Saulcy, F-57045 Metz Cédex 1, France

Abstract
CIMOSA is an open systems architecture for CIM developed by the ESPRIT Consortium AMICE. It structures a CIM system as a set of concurrent communicating processes executed by a finite set of functional entities. Modelling is based on a three-stage, process-based enterprise modelling approach (for business requirements definition, system design specification and implementation description). This paper presents the CIMOSA process model. It covers functional/behavioural, information, resource and organisation aspects of an integrated enterprise at each modelling level. The approach enforces the genericity and reusability principles by the use of generic building blocks to build partial models and particular models.

Keywords
CIMOSA, Enterprise Integration, Enterprise Modelling, Business Process Modelling, Workflow

1 INTRODUCTION

Enterprise modelling is a prerequisite to enterprise integration (Vernadat, 1996). Enterprise integration (EI) is concerned with breaking down organisational barriers and harmonising business process execution within an enterprise or the extended enterprise to improve efficiency by creating a synergistic whole. Enterprise modelling is also a central activity in Business Process Reengineering (BPR). BPR consists of simplifying enterprise processes to reduce excessive delays or costs in the enterprise operations.

Various approaches have been proposed for enterprise modelling. Among these, we can mention the IDEF suite of models (ICAM, 1981) or the GRAI-GIM method (Doumeingts et al., 1992) as early methods which were strongly activity-based. Since then, modern methods have adopted a process-based approach such as CIMOSA (AMICE, 1993; Kosanke et al., 1995; Vernadat, 1993), ARIS (Kruse and Scheer, 1994) or IDEF3 (Mayer et al., 1992) to more adequately model enterprise behaviour, and especially business processes.

This paper presents the principles of the CIMOSA process model, and especially the workflow constructs. Its practical use has been described by Zelm et al. (1995).
2 CIMOSA PRINCIPLES

CIMOSA, the European Open Systems Architecture for CIM, has been developed by the AMICE Consortium as a series of ESPRIT Projects (EP 688, 5288 and 7110) over 7 years. It was jointly financed by the European Commission and project partners (30 companies in total putting together CIM users, CIM vendors and a few academic organisations).

CIMOSA is made of (AMICE, 1993):
• a Modelling Framework based on a process model,
• an Integrating Infrastructure consisting of information technology (IT) services to support enterprise integration, application interoperability and model execution, and
• a methodology, called CIMOSA Modelling Process, which follows the CIM system life cycle. It derives (1) a design specification model and (2) an implementation description model which must both comply with the business requirements definition model defined first for the enterprise.

CIMOSA considers the enterprise as:
• a set of communicating concurrent processes governing the execution of elementary actions called functional operations,
• a finite set of agents, called functional entities, executing the functional operations required by business processes and processing enterprise objects.

Functional entities are any resource entities capable of performing actions, i.e. functionalities. Functional entities range from programmable agents to intelligent autonomous agents. Examples include NC machines, robots, computers, software applications, database servers or human operators.

The link between functional entities and business processes is made by functional operations which are the atoms of functionality in the model. A functional operation is any message (request or data) sent or received by an agent. It is defined by a name and list of arguments. Examples are MMS commands for NC machines, SQL statements for database servers, API primitives for specialised software packages, instructions to operators.

Enterprise objects are entities used, transformed, produced or consumed by functional entities of the enterprise. They are characterised by a set of properties. They exist in the real-world under different appearances or states, called object views. Two types of object views are distinguished in CIMOSA: physical object views (representing physical appearances of objects) and information object views (data representations of objects). This distinction allows to separately model the material flow and the information flow.

Processes model the control flow. They are triggered by events and can be structured into basic steps called enterprise activities. An enterprise activity is a task to be performed by one or more functional entities allocated to the activity for the whole duration of the task. A task is a sequence of functional operations. CIMOSA distinguishes two types of processes: core processes or end-to-end processes called domain processes and sub-processes called business processes.

Because there is a huge number of processes and functional entities in an enterprise, CIMOSA introduces the concept of domain to manage system complexity. A domain is functional area of an enterprise grouping domain processes and functional entities contributing to some common business objectives. A domain does not necessarily map a department of an enterprise. It is a user-defined set of domain processes.

This paradigm is illustrated by Fig. 1 and Fig. 2. Figure 1 shows the functional decomposition of domains into domain processes and then domain processes into business processes and enterprise activities. It also illustrates that processes model enterprise behaviour (by means of behavioural rules) while enterprise activities model enterprise functionality (inputs, outputs, transformation). Figure 2 illustrates the decomposition of activities into functional operations and the cross-reference between functional operations (i.e. required functionality) and functional entities (i.e. provided functionality).

We claim that the paradigm presented in Fig. 1 is sufficient for BPR studies because in BPR the focus is on the flow of control and not on resource assignment. Necessary constructs are explained next.
3 CIMOSA PROCESS MODEL

The modelling approach underlying CIMOSA is an event-driven, process-based model (Vernadat, 1993). It allows to describe either highly non-deterministic event-driven systems (governed by message exchange) or deterministic rule-driven systems or hybrid systems (event-driven and rule-driven). The modelling language is defined as a set of constructs, called generic building blocks (GBB's), and defined as object classes. The following GBB's have been defined in CIMOSA (AMICE, 1993):

**Domain**
This is a structuring construct which allows the user to organise his/her model into autonomous functional sub-areas to manage system complexity. A domain contains domain processes. It is defined by:
- its name and identification
- its scope defined in terms of business objectives
- the full list of domain processes contained in it
- relationships with other domains in terms of events and object views exchanged
Object View
An object view depicts a real-world manifestation of one or more objects. It is a projection of some objects over some of their properties at a certain time. It therefore represents a state of an object or a state derived from several objects. Only object views are manipulated by functional entities executing activities, not enterprise objects themselves. Enterprise objects are characterised by their properties (i.e. attributes) and are structured into object classes using the generalisation/specialisation (is-a link) and aggregation (part-of) abstraction mechanisms. Object views are defined by:
- name and identification
- a reference to related enterprise objects
- list of properties, i.e. object properties or object views

Event
An event represents any solicited or unsolicited happening (e.g. order, request, machine failure, or end/start of action) of the real-world that triggers a chain of activities, i.e. a domain process. Events can be generated by: external sources (coming from outside the domain or the enterprise), by functional entities or by enterprise activities. They are defined by:
• a name and identification
• their source
• the process(es) they contribute to trigger
• a predicate defining their occurrence condition (design level)
• an object view identifying an object attached to the event (optional)

Domain Process
A domain process is a full, i.e. end-to-end process of the enterprise. It can be decomposed into sub-processes and is made of enterprise activities. It is triggered by one or more events. It depicts some enterprise behaviour by means of a behavioural rule set defining its control flow as WHEN condition DO action rules It is defined by:
• a name and identification
• its (non-empty) list of triggering events
• its process behaviour defined as a set of behavioural rules

Business Process
Nearly identical to a domain process except that it must be called by a parent structure and is not directly triggered by events only.

Enterprise Activity
Enterprise activities model enterprise functionalities. An enterprise activity is the location of action (transformation or decision) in the model. It uses input objects states and produces output object states (defined as object views) using some transfer function. Its execution is summarised by a termination status called ending status (defined as a 0-argument predicate). It has a duration and needs resources having the necessary required capabilities. The transfer function or activity behaviour is defined in terms of the functional operations mentioned earlier (atoms of functionality). They are defined by:
• a name and identification
• function input and output defined as object views
• control input defined as information object views and control output defined as ending statuses
• resource input (a set of functional entities) and resource output (information object view on resources used)
• a transfer function or activity behaviour defined as an algorithm for transformation activities, a script for human-based activities, expert systems for computer-based decision-making
• minimal, mean and maximum duration
• required capabilities (if selection of required resources is made on-the-fly)
• the complete set of possible ending statuses

Resource
Resources can be of two types: passive resources (not able to execute functional operations by itself such as a tool, a cart, etc.) and active resources having a control device (and being able to execute functional operations sent as commands). Active resources are the functional entities introduced to earlier. Resources are defined by:
• a name and identification
• an object view defining their characteristics and referring to the enterprise object representing them
• a set of provided capabilities
• a status and availability
• the set of functional operations that they can execute (for functional entities only)
• their list of component resources in the case of compound resources
Organisation Unit
An organisation unit is an enterprise object (usually a resource) assuming responsibility and authority on one or more elements of the model as described by the previous constructs. For instance, a foreman can be the supervisor of two machining cells (defined as resources) performing some activities of process plans (defined as business processes of a domain process) of some products (defined as object views on product enterprise objects).

Organisation Cell
This construct is used to structure organisation units into larger entities at different responsibility levels in order to model the organisation structure of the enterprise. An organisation cell must be placed under the responsibility of an organisation unit defined as a human operator.

Other generic building blocks exist in CIMOSA such as Objective/Constraint, Domain Relationship, Declarative Rule, Capability Set, Enterprise Object and Object Relationship (AMICE, 1993). They can be used at the various modelling levels of CIMOSA (requirements definition, design specification, implementation description) but are not described here for the sake of conciseness.

Behavioural rules
The following rules are used to describe the flow of control of structured processes:

- Process triggering rules: They are used to start a process by means of events.
  WHEN (START WITH event-i) DO EFI
  In this case, the process starts with function EFI any time that an occurrence of event-i occurs.

- Forced sequential rules: They are used when a function EF2 must follow another function EF1 whatever the ending status (given by the function ES) of EF1 is. The reserved word 'any' is used in this case (not an ending status).
  WHEN (ES(EF1) = any) DO EF2

- Conditional sequential rules: They are used to represent branching conditions in the flow of control.
  WHEN (ES(EF1) = end_stat_1) DO EF2
  WHEN (ES(EF1) = end_stat_2) DO EF3
  WHEN (ES(EF1) = end_stat_3) DO EF4

- Spawning rules: They are used to represent the parallel execution of enterprise functions in a flow of control. Two types of spawning rules can be defined:
  (a) Asynchronous spawning: When EF1 is finished with status 'value', EF2, EF3 and EF4 are requested to start as soon as they are ready (& is the parallel operator).
    WHEN (ES(EF1) = value) DO EF2 & EF3 & EF4
  (b) Synchronous spawning: When EF1 is finished with status 'value', EF2, EF3 and EF4 are all requested to start exactly at the same time (SYNC indicates the synchronisation).
    WHEN (ES(EF1) = value) DO SYNC (EF2 & EF3 & EF4)

- Rendez-vous rules: They are used to synchronise the end of spawning rules.
  WHEN (ES(EF2) = value_2 AND ES(EF3) = value_3
  AND ES(EF4) = value_4) DO EF5

- Loop rules: They are used to execute again the same type of enterprise function.
  WHEN (ES(EF1) = loop_value) DO EF1

- Process termination rules: They are used to indicate the end of the process.
  WHEN (ES(EF2) = end_stat_x
  AND ES(EF2) = end_stat_y) DO FINISH

Using these rules, a process behaviour is said to be consistent if FINISH can be reached from all STARTs and all enterprise functions used in the rules belong to at least one path from START to FINISH.

Ill-structured (or semi-structured) processes are processes for which the exact sequence of all employed activities and/or sub-processes is not completely known. In this case, temporal logic (Allen, 1981) can be used and the process behaviour becomes:
Process Behaviour: <behavioural rules>
[Subject To <temporal rules>] End Process

Two types of behavioural rules are added to the previous set to model ill-structured processes: random choice rules and unordered set rules. In these rules, the action part is reinterpreted (variable S), meaning that it is considered as a whole to make possible the definition of its ending status.

- Run-time choice rules: These rules are used when there is an exclusive choice among several alternatives. Exactly one function in the list will be executed as decided by the resource at run-time, which must be common to all functions in the list.
  WHEN (ES(EF1) = end_stat_1) DO S = (EF2 | EF3 | EF4)

- Unordered set rules: These rules say that a set of functions must be executed next but the order is unknown. In this case, all functions are supposed to be executed.
  WHEN (ES(EF1) = end_stat_1) DO S = {EF2, EF3, EF4}

The flow of control for the set of functions of the unordered set rules can be constrained by the following temporal rules in the "subject to" clause:
- A Before B: A must be executed sometime before B
- A Meets B: B starts just after A ends
- A During B: A is executed after B starts but during B
- A Starts B: when A starts, B must start
- A Finishes B: when A ends, B must end

The complete flow of activity execution of a process is only known upon completion of the process (as well as its ending status). This flow of activities over time is called the trace of the process.

4 EXAMPLE

Let us consider a simple design environment for electro-mechanical components. It concerns an engineering process for designing and testing a product made of a mechanical system and an electrical system (Fig. 3).

The process starts with a conceptual development activity. When it is done the process continues with two parallel sub-processes, one for the mechanical system design and one for the electrical system design. If both sub-processes finish successfully (ending statuses MSD-OK and ESD-OK respectively), the process goes for product prototype and then for test. If the test is not successful (ending status NOK), the process goes back to conceptual development. This also happens if either mechanical system design or electrical system design encounters
problems (ending statuses MSD-pb and ESD-pb respectively). If the test is successful then the process continues for manufacturing and finally ends. The process is specified as follows:

**DOMAIN PROCESS** Product_Design  
**TRIGGERING EVENTS:** Design_Order  
**PROCESS BEHAVIOUR:**
- WHEN (START WITH Design_Order) DO Conceptual_Development
- WHEN (ES(Conceptual_Development) = done)
  - DO Mechanical_System_Design & Electrical_System_Design
- WHEN (ES(Mechanical_System_Design) = MSD-pb)
  - DO Conceptual_Development
- WHEN (ES(Electrical_System_Design) = ESD-pb)
  - DO Conceptual_Development
- WHEN (ES(Mechanical_System_Design) = MSD-OK AND ES(Electrical_System_Design) = ESD-OK) DO Product_Prototype
- WHEN (ES(Product_Prototype) = any) DO Test
- WHEN (ES(Test) = OK) DO Manufacturing
- WHEN (ES(Test) = NOK) DO Conceptual_Development
- WHEN (ES(Manufacturing) = done) DO FINISH

END PROCESS

Let us now assume that the sub-process Electrical_System_Design is an ill-structured process made of three basic activities: Rotor_Design, Stator_Design, Body_Design. Some designers may prefer to start with Rotor_Design, others with Stator_Design. We can specify the Electrical_System_Design process as follows:

**BUSINESS PROCESS:** Electrical_System_Design  
**TRIGGERING EVENTS:**  
**ENDING STATUSES:** ESD-OK, ESD-pb  
**PROCESS BEHAVIOUR:**
- WHEN (START) DO A = \{Rotor_Design, Stator_Design, Body_Design\}
- WHEN (ES(A) = completed) DO FINISH  
SUBJECT TO
  - Rotor_Design BEFORE Body_Design;  
  - Stator_Design BEFORE Body_Design

END PROCESS

5 CONCLUDING REMARKS

The CIMOSA process model for business process modelling has been described. The model enforces an event-driven, process-based approach using generic constructs. These constructs can be specialised or customised to build partial or particular models (Zelm et al., 1995). The paper is intended as a complement to (Vernadat, 1993). Since then, most of the constructs presented in this paper have been discussed and approved in the work of the CEN/TC 310/WG1 group of the European standardisation committee on constructs for enterprise modelling and integration. The model has been one of the major input sources for the definition of a new prenorm ENV 12 204 adopted and published at the end of 1995 (CEN, 1995).
6 REFERENCES


Kurt Kosanke received an engineering degree in Physics from the Physikalisch Technische Lehranstalt in Lübeck, Germany. He joined the Development Laboratory of IBM Deutschland in Böblingen, Germany, working on advanced development projects on optical printers and large-scale printers using laser and light deflection technologies. After a four year assignment to IBM USA he joined the manufacturing research organisation of IBM Deutschland. K. Kosanke has held management positions in the areas of instrument development, production control, material logistics and simulation. Since 1984 he has been responsible for IBM involvement in ESPRIT and similar programs. He has been the director of the ESPRIT Consortium AMICE and he is now the director of the CIMOSA Association e.V. promoting CIMOSA and Enterprise Integration in industry. Kurt Kosanke holds many patents in most of the working areas mentioned above and has published numerous papers in these fields as well.

François Vernadat is a French and Canadian citizen. He obtained his Master degree in Electronics and Automatic Control in 1977 and the Ph.D. degree in 1981 from the University of Clermont, France. From 1981 till 1988, he has been a researcher officer at the Division of Electrical Engineering of the National Research Council of Canada, Ottawa, Canada. In 1988, he joined INRIA, a French research institute in computer science and automatic control. He is currently a professor at University of Metz, France. His research interests include CIM, database technology and information systems, enterprise modeling and integration, knowledge representation, formal description techniques, Petri nets, and model enactment. Beside his work on the M* methodology for CIM information systems, he has headed the development of a main-memory database system (DBS/R), and was one of the chief architects of CIMOSA, an Open System Architecture for CIM initially developed as an ESPRIT project (AMICE). He has authored and co-authored over 100 scientific papers in journals, conferences, and books. He is the co-editor of the books “Advanced in Factories of the Future, CIM and Robotics” (Elsevier), “Practice of Petri nets in Manufacturing” (Chapman & Hall) and "Enterprise Modeling and Integration: Principles and Application" (Chapman & Hall). He is the European editor for the International Journal of Computer Integrated Manufacturing. Prof. Vernadat is a member of the IEEE Computer Society, ACM, and SME.

Martin Zelm holds a Ph.D. degree in Physics from Hamburg University. He is now an independent consultant and founding member of the CIMOSA Association. His main work area is consulting in the development of enterprise integration and related information dissemination. Until 1993, Dr. Zelm was a manager of the ESPRIT projects department at IBM Germany, where he specialised in CIM architectures. He has contributed to the design of the CIMOSA specifications for enterprise modelling and to the European pre-standard ENV 40003 "Framework for CIM system integration". Dr. Zelm is the author of a number of technical papers, mostly about enterprise modelling with CIMOSA.
A Formal Requirements Engineering Framework for CIM Infrastructures Reengineering

Michaël Petit and Eric Dubois
Computer Science Department, University of Namur
21 rue Grandgagnage, B-5000 Namur, Belgium, \{mpe,edu\}@info.fundp.ac.be

Abstract
The complexity of the task of designing Computer Integrated Manufacturing (CIM) Systems that can evolve with time is recognized. This complexity definitively makes a methodological approach necessary. The paper proposes a methodology for the task of designing Computer Integrated Manufacturing (CIM) Systems that can evolve with time. It is based on requirements engineering formal languages that enable the expression of (i) the functional goals of the CIM system, (ii) the quality goals on the system (what qualities the system should have), (iii) the specification of alternatives solutions that fulfill the functional goals and (iv) the evaluation of these alternatives w.r.t. the quality goals. To support this methodology we introduce: the Albert II language (for the description of the functional goals) and the i* framework (for the description of the rationales underlying a system design).

Keywords
Functional and Quality Requirements, Business Process Reengineering, Formal Specification Language

1 INTRODUCTION
The design of Computer Integrated Manufacturing (CIM) Systems is recognized as a complex task because the complexity of the built systems themselves. This complexity is due a.o. to the large number of components they contain, the diversity of these components (hardware, software or humans) and the complexity of the data involved (e.g. CAD drawings of parts, workschedules, …) Moreover, CIM systems have to cope with organisational and technological changes that happen during their lifetimes. Methodological approaches are made necessary because of the amount of investment that these systems represent and because errors in the choice and
implementation of a system may have tremendous financial consequences. This paper proposes a methodology aimed at helping part of the design process and the maintenance of CIM systems.

The process of designing a system is generally decomposed in two main phases: (i) the requirements engineering (RE) activity consists in eliciting and documenting the requirements that users put on the system to be built and its own constituents (devices, software or humans) and (ii) the design engineering (DE) activity consists in the further development and implementation of these constituents. During the RE phase, a number of decisions are made with respect to e.g. the assignment of tasks to humans, devices and software components. These choices are part of a general system design activity (as opposed to the DE activity introduced above).

In this paper, we present research work addressing aspects of the RE activity. The proposed RE methodology uses a combination of several languages and models in order to help decisions made during the system design process and keep a trace of them for further system maintenance and evolution. We illustrate on a toy CIM case study how the methodology can be used to re-engineer an existing system and justify the transition to a slightly modified system. The methodology, applied for the evolution of a system consists in the following steps:

1. Re-modeling of the behaviour of the existing system using a formal specification language.
2. Eliciting and/or re-modeling of the organizational structure and initial rationales for the choices made when first designing the system and formal expression of the functional goals identified in the organizational model in order to verify if they are still currently met in the existing system.
3. Modification of the organizational model to reflect the current situation in terms of goals, their satisfaction and current priorities among them, formalization of the modified functional goals and modeling of alternatives for satisfying the new goals and of the rationales for the choice of the new solution.
4. Formal modeling of the chosen solution (to be implemented).

Phases 1 and 4 of the methodology rely on the use of a formal language called \textit{Albert II}. The language is briefly described in Section 2 and illustrated with fragments of the specification of a simplified CIM toy system. Phases 2 and 3 are based on the i* framework developed at the University of Toronto. This framework is introduced in Section 3 and illustrated through the handling of the case study. The evolution of goals, some alternatives to the fulfillment of these and the rationales for the choice among them, are presented in Section 4. The final \textit{Albert II} specification of the new system to be implemented is not presented in this paper due to the lack of space. All along the paper, we use a CIM toy case study to illustrate the proposed methodology. The informal description of this case study follows.

A production cell, under the responsibility of a manager is made of a number of workstations (WS). Each WS can remove a part from its input buffer and transform it into another part which is placed in its output buffer. All buffers can contain no more than one part at any time.

An AGV system is used for the transportation of parts between the WS's. We concentrate on the transportation between 2 WS: WS1 and WS2. Part of the goal of the AGV system is to transport each part made available in the output buffer of WS1 to the input buffer of WS2. The AGV system is composed of a controller and an AGV. The controller has knowledge about the status of the buffers and of the status of the AGV (busy or not). On the basis of this information, it issues orders to the AGV for both removing the part from a WS and delivering it to another WS.
2 THE \textbf{Albert II} LANGUAGE

The \textit{Albert II} specification language is a formal requirements specification language. Its design started around 1992 and the validation of the language constructs has been done through its use in the context of non-trivial systems like Computer Integrated Manufacturing (Dubois et al., 1993) or telecommunication systems (Dubois et al., 1995). Like other formal languages proposed for requirements engineering, \textit{Albert II} is characterized by its expressiveness and its formality.

- \textit{Albert II} allows the expression of functional requirements inherent to real-time distributed (cooperative) systems, and
- the semantics of the language relies on some object-oriented variant of real-time temporal logic called \textit{Albert-CORE} (Du Bois, 1995).

The essential feature of \textit{Albert II} is however its \textit{naturalness}, i.e. the possibility to map informal statements provided by customers in a straightforward way onto formal \textit{Albert II} statements. The use of \textit{Albert II} allows one to avoid the introduction of extra elements in the formal specification which would have no counterpart in customers statements. This means that \textit{Albert II} is well-suited to the establishment of traceability between informal and formal requirements specifications. It also means that \textit{Albert II} specifications can be validated by rephrasing them into a natural language text understandable by customers.

In \textit{Albert II}, functional requirements are expressed in terms of a set of formal statements in typed temporal first order formulas. In order to enhance readability, a specification is organized into units called \textit{agents}. Logical statements are grouped around agents in order to define the set of admissible behaviors (or \textit{lives}) they may experience. Logical statements describing an agent are classified into categories, each corresponding to a pattern of property. Such pattern provides guidance in the elicitation and structuring of requirements.

The language is made up of (i) a graphical component for \textit{declaring} the vocabulary of the application to be considered and (ii) a textual component for \textit{constraining} the admissible behaviors of agents through logical formulas. For a detailed presentation of \textit{Albert II}, see (Du Bois, 1995).

\textit{Declarations}

The Declarations component consists of a description of the general structure of the composite system in terms of agents as well as of the structure of each individual agent.

A specification consists of a collection of agents. Figure 1 shows the structure of the \textit{Cell}. \textit{Transportation-System} is an aggregate of one \textit{Transp-Controller} and one \textit{AGV} whereas \textit{WS} is a population (class) of agents.
The declaration part of an agent consists of the description of its state structure (i.e. the memory of the agent) and the list of actions which may happen during the life of the agent and which may change the state of the agent. State components (graphically depicted with rectangles) are typed and actions (graphically depicted with ovals) can have typed arguments. Types may vary from simple data types to complex data types (recursively built using predefined type constructors).

In the example (see Figure 2), the state of the AGV agent is structured into two individual values (resp. Busy of type BOOLEAN and Position of type WS) and a set of parts (Contents). The AGV may perform five actions: Load, Unload, Moveto, Get and Put. A Get action, e.g. has an argument of type PART which indicates the part transferred on the AGV and an argument of type WS which indicates the identity of the WS where the part is being gotten.

In addition, the graphical notation also expresses visibility relationships linking agents to the outside. Arrows on Figure 2 show how agents make information visible to other agents, e.g., the Busy value of the AGV agent is exported to the Transp-Controller agent: on the contrary, the Position value is shown to no other agents. Arrows also show how agents influence each others’ behavior through exportation of actions, e.g., the AGV agent is influenced by the RequestGet actions of the Transp-Controller agent.

**Constraints**

Constraints are used for pruning the (usually) infinite set of possible lives associated with the agents of a composite system. The life of an agent is (usually) an infinite alternating sequence of changes (occurrences of actions) and states values. An admissible life will respect:

1. local constraints related to the internal behavior of the agent;
2. cooperation constraints defining how the agent interacts with other agents.

Local constraints are classified under five headings. The use of four of them is illustrated in the example.

**Effects of Actions.** The effect of an action is expressed through its functional characterization in terms of a mathematical relationship between successive information states (see, e.g. on Figure 3, the effects of the Load and Unload actions)

**Action Composition.** Composition rules express restrictions on admissible sequences of actions. They also allow to introduce composed actions made of more finer actions. On Figure 3, the illustration of this concept of process is given by the introduction of e.g. a Get action. This
LOCAL CONSTRAINTS

STATE BEHAVIOUR

Card(Contents) ≤ 1

The AGV can contain maximum one part.

Card(Contents) ≠ 0 ⇒ Busy

If the AGV contains a part, then it is busy transporting this part.

EFFECTS OF ACTIONS

Load(p,pos): Contents := Add(Contents,p)

The action of loading a part adds this part to the contents of the AGV.

Unload(p,pos): Contents := Remove(Contents,p)

The action of unloading a part removes this part from the contents of the AGV.

Moveto(pos): Position := pos

At the end of a moveto action, the AGV arrives at a particular WS.

Transp-Controller.GetRequest(p,pos): Busy := TRUE

Put(p,pos): Busy := FALSE

The AGV is busy since the moment it receives a get request from the controller until it finishes the corresponding put action.

CAPABILITY

F (Load(p,pos)/pos ≠ Position ∨ p ∈ pos.Output) ∨ Card(Contents) ≠ 0)

The AGV can only load at a WS where it lies a part which is present in the output buffer of that WS if it does not already contain a part.

F (Unload(p,pos)/pos ≠ Position ∨ p ∈ Contents ∨ Card(pos.Input) = 0)

The AGV can only unload a part at a WS where it lies if it contains the part and if the input buffer of the WS is empty.

F (Moveto(p)/Position = p)

The AGV can not move to a position that it already occupies.

ACTION COMPOSITION

Get(p,pos) ↔ Transp-Controller.GetRequest(p,pos); (Moveto(pos) @ dac); Load(p,pos)

A Get process is initiated by a request of the controller, implies an eventual movement to the specified WS and a load action for the part at that WS.

Put(p,pos) ↔ Transp-Controller.PutRequest(p,pos); (Moveto(pos) @ dac); Unload(p,pos)

A Put process is initiated by a request of the controller, implies an eventual movement to the specified WS and an unload action for the part at that WS.

Figure 3 Constraints on the AGV agent.

...
COOPERATION CONSTRAINTS

ACTION PERCEPTION

\( \mathcal{X} \mathcal{K} \left( \text{Transp-Controller}.\text{GetRequest}(p,\text{pos}) / \neg \text{Busy} \right) \)

Get requests from the controller are only taken into account when the AGV is free (not busy).

\( \mathcal{K} \left( \text{Transp-Controller}.\text{PutRequest}(p,\text{pos}) / \text{TRUE} \right) \)

Put requests from the controller are always taken into account.

STATE PERCEPTION

\( \mathcal{K} \left( \text{WS}.\text{Output} / \text{TRUE} \right) \)

\( \mathcal{K} \left( \text{WS}.\text{Input} / \text{TRUE} \right) \)

The status of the output buffer of WS1 and of the input buffer of WS2 are always known by the AGV.

ACTION INFORMATION

\( \mathcal{K} \left( \text{Get}(p,\text{pos}),\text{WS} / \text{pos} = \text{WS} \right) \)

\( \mathcal{K} \left( \text{Put}(p,\text{pos}),\text{WS} / \text{pos} = \text{WS} \right) \)

Put and get actions are only shown to the WS where they occur.

STATE INFORMATION

\( \mathcal{K} \left( \text{Busy},\text{Transp-Controller} / \text{TRUE} \right) \)

The status of the AGV is always shown to the controller.

\( \mathcal{K} \left( \text{Contents},\text{Transp-Controller} / \text{TRUE} \right) \)

The contents of the AGV is always shown to the controller.

**Figure 4** Constraints on the AGV agent (Continued).

invariants) or dynamic constraints (on the evolution of the state). The latter are expressed using temporal connectives. See examples of these constraints on Figure 3.

Cooperation constraints are classified under four headings describing how an agent perceives action performed by other agents (Action Perception), how it can see parts of the state of other agents (State Perception), how it lets other agents know of the actions that it does (Action Information) and how it show parts of its state to other agents (State Information). Perception and information provide the analyst a way to add a dynamic dimension to the importation and exportation relationships between agents expressed in the declaration part of the specification. The headings are illustrated on Figure 4, e.g., the first Action perception constraint defines the conditions under which the AGV agent is influenced by GetRequest actions of the Transp-Controller (in this case, if and only if the AGV is not busy).

3 ORGANIZATIONAL MODELS: THE i* FRAMEWORK

In this section, we introduce and illustrate the use of the i* framework for gaining a deeper understanding about the organizational environment, helping to explore alternative patterns of relationships (among software, devices and human components), discovering the implications of these alternatives for each agent, and helping to make tradeoff among the alternatives.

The i* framework provides understanding of the ‘why’ by modeling organizational relationships that underlie system requirements. Agents are taken to have goals, and use knowhow and
resources in their attempts to achieve goals. The framework includes two models. In the Strategic Dependency model, agents are modeled as depending on each other for goals to be achieved, tasks to be performed, and resources to be provided. In the Strategic Rationale model, the reasoning that each agent has about its relationships with other agents are described. It supports reasoning about alternative ways for meeting goals, and for evaluating them. Agents are strategic in that they are concerned about opportunities and vulnerabilities.

In the sequel, the $i^*$ framework will not be presented in detail. Such presentation can be found in different publications where it is applied in other contexts than CIM (e.g. (Yu and Mylopoulos, 1994) and (Yu et al., 1995)). We will stress on the kind of information conveyed by the framework and suggest how it is complementary to information collected in the Albert II specification.

3.1 The Strategic Dependency Model

The Strategic Dependency Model (SDM) supports the description of organizational environments by showing external relationships among actors. On Figure 5, in the context of the case study, such relationships are denoted by labeled links going from one agent to another. As depicted, these agents are those identified in the Albert II specification but, with the exception of the Manager who is an additional agent playing a role at the organisational level. The SDM provides four types of dependency links used to differentiate among the kinds of intentional relationships existing between depender and dependee agents. The use of three of them is illustrated in Figure 5.

In a goal dependency, one agent depends on another to bring about a condition in the world. The depender does not care in what way the dependee accomplishes the condition. In our case study, we have identified a goal dependency called Have parts transferred(WS1,WS2) which expresses that, within the system, it is the hope of the Manager that the Transp-Controller agent will act in such a way so that parts will be transferred from WS1 to WS2. Two other goal dependencies are Begin Transfer by AGV and End Transfer by AGV which indicate that the Transp-Controller depends on the AGV for ensuring the transportation but does not care about how it accomplishes it. Goal dependencies of the $i^*$ model are related to some properties associated with the Albert II specification. More precisely, a goal can be associated with theorems expressed by introducing global constraints on the action occurrences and state components evolution. For example, the Have parts transferred(WS1,WS2) goal can be formalised as follows:

\[
(p \in WS1.\text{Output} \implies \diamond p \in WS2.\text{Input})
\]

Any part \(p\) which is in the output of WS1 must be some times latter \(\diamond\) in the input of WS2.

Since the fulfillment of this goal is depending on the Transp-Controller, we will check its incidences at the level of the Strategic Rationale Model associated with the Transp-Controller (See Section 3.2).

In a resource dependency, the depender depends on the dependee for the availability of an entity (physical or informational). This resource is considered as the finished product of some process followed by the dependee for some other purposes. In the case study, Output is modeled as resource dependency. This means that the Transp-Controller is expected to know about the contents of the output buffer associated with WS1. The existence and the visibility of the output
buffer is not related to *Transp-Controller* objective of accessing it but is an intrinsic feature of the *WSI*. That is why it is related with a resource dependency which allows to trace the information that if the properties of *WSI* were changed in the future, the existence or not of the output buffer would influence the behaviour of the *Transp-Controller*. Again, it is possible to relate the resource goal with some fragments of *Albert II*. In the specification of *WS* (not given in the paper), such fragments are:

\[
\text{Output set of PART} \rightarrow \text{CELL} \\
\kappa (\text{Output}_x / \text{TRUE})
\]

The status of the output buffer is always shown to all agents of the cell.

In the case study, other resource dependencies are *Input* (i.e. the contents of the input buffer associated with *WSI*), *Busy* (i.e. the status of the *AGV*) and *Contents* (i.e. the set of parts located on the *AGV*).

Figure 5 shows also *softgoal dependencies*: *Transport Efficiency*, *Cost* and *Flexibility*. A softgoal dependency is similar to a goal dependency except that the condition is not sharply defined a priori. For example, the *Manager* depends on the *Transp-Controller* for the transport from *WS1* to *WS2* to be efficient. What is 'efficient' is a matter of interpretation. While the *Transp-Controller* may do its best to ensure the efficiency of the transport, it is the manager who decides whether it is efficient enough for his purposes. Softgoal dependencies have no direct counterpart in *Albert II* specifications. However, they heavily influence the design decisions occurring during the elaboration of the solution described by the specification. The nature of these decisions and their impact with respect to softgoal dependencies are further detailed in the next subsection.

Finally, let us mention that there is a fourth kind of link called *task dependency* in the *i*\textsuperscript{*} framework. This use of this link, not illustrated in our case study, allows to indicate that the depender depends on the dependee to perform an activity. A task dependency specifies *how* the task is to be performed, but not *why*. 

Figure 5 A Strategic Rationale model for the cell example.
3.2 The Strategic Rationale Model

Whereas the Strategic Dependency model gives an external view of how agents depend on each other, the Strategic Rationale Model (SRM) gives a more detailed description of the rationales behind the dependencies. One can answer ‘why’ questions more precisely. Like in the SDM, intentional elements (goal, resource, task, softgoal) appear in the SRM but are complemented with specific links associated with means-ends relationships and task-decompositions.

In the SRM associated with the Transp-Controller agent depicted in Figure 5, the external goal to Have Parts Transferred(WS1,WS2). is met by the Transfer Parts task which indicates that a certain procedure is followed by the Transp-Controller to this end. This task consists of (represented via task-decomposition links): controlling the transfer, accessing to the content of the output buffer of WS1 and to the content of the input buffer of WS1. Control Transfer appears as a goal to be met. This means that freedom is left to the designer of the system about choosing the way to meet it. In our case, it has been decided that this goal can be met (represented via a means-ends link) by controlling an AGV in charge of the transfer. This Control AGV Transfer task is again decomposed in two goals and two resources dependencies.

As we mentioned, several alternatives could have been imagined for meeting the Control Transfer goal. The evaluation of the different possibilities as well as the selection of a specific one have to be related to the identified softgoals. In the case study, we have already noticed that the manager wants a transportation system being efficient, cost effective and flexible. Different ways of transferring parts (i.e., different organizational configurations) may be evaluated as contributing positively or negatively to these goals. In our case, using an AGV system is considered to be good (or at least sufficient) for the efficiency (tested by the means of simulations). The flexibility* for the AGV is also considered to be good because of its ability to link easily any two workstations in both directions. Cost-effectiveness is also good since only one transporter is needed to fulfill the transportation requirements between all WS of the cell, so the hardware costs were limited and compensated the higher investment necessary for the AGV controller.

Like for the SDM, elements of the SRM model can be linked to fragments of the Albert II specification. With respect to the formal specification of the Transp-Controller given in the annex, the following links can be established:

- With the Transfer-Parts task are associated fragments of Albert II specification related to the perception of the content of the input and output workstations buffers and the identification of the Transport action (see statements 1 and 10).
- With the Control Transfer goal are associated specifications related to the conditions under which the transfer can occur (see statements 4-7).
- With the Control AGV Transfer task are associated specifications associated with the decomposition of the Transport action in terms of finer actions controlling the AGV (see statements 2, 3, 7, 9 and 11)

Finally, on Figure 5, one can also see the SRM associated with the AGV. Typically, at its level, only tasks are identified since AGV’s have ‘hard-coded’ behaviours which do not offer flexibility in their design (viz no intermediate goals can be identified).

*When evaluating flexibility, different facets of it can be envisaged. We consider here only one of them: the number of possible ways to perform the same tasks (in our case, alternative routings of parts between WS that produce the same end-results).
4 CHANGES MANAGEMENT AND CONCLUSIONS

Basically, if we are spending time and effort to capture all the requirements (both the ‘whats’ and the ‘whys’) underlying a system, this is because we are convinced that all this information will be of a great interest when the system will have to evolve and be maintained in the future. If we aim at building real flexible systems, we need to offer facilities for changes management, in particular at the requirements engineering level.

Hereafter, we examine how the case-study system could evolve in response to a change in the load of the cell (functional aspects) and in the priority among soft goals (non-functional aspects). We also indicate which information is required for managing this change.

Let us suppose that a change in the load of the cell has increased the transport requirements (in quantity) between WSI and WS2. As discussed in Section 3, the current AGV system was initially considered as satisfactory with respect to efficiency, flexibility and cost-effectiveness. But now, the efficiency of the transport system is too unsatisfactory since most of the time, the part produced by WSI has to wait in the buffer before to be transported to WS2. This waiting time causes delay to WS1 because no other part can be produced while the buffer is occupied. Several alternative solutions to this problem have been imagined. Only one of them is described here.

A solution where a conveyor is used for only the transportation of parts from WSI and WS2 and where the AGV remaining responsible for the rest of the transports in the cell was imagined. The Albert II specification is not given in this paper. A new i* SR model is elaborated to compare both alternatives and contains the following information.

The conveyor solution is certainly worse than the AGV w.r.t. cost since the AGV system is retained (and slightly modified) and an extra investment has to be engaged for the implementation of the conveyor. The flexibility of the cell (as defined above) is worsened since with the AGV system, though this possibility was not used, parts could be easily transported from any WS to any other WS. With the conveyor system, there exists a fixed link from WSI to WS2 limiting the flexibility of transport. The efficiency is far better for several reasons: parts in WSI do not have to wait for the AGV to come and load them anymore, the AGV now has to handle less transport requests than before and the efficiency of WSI is better because the conveyor acts as a buffer between WSI and WS2. This important increase in efficiency compensates for the drawbacks of this solution. This reflects a change in the priority among soft-goals.

The last two steps of our methodology consist in the elaboration of the specification of the new system to be implemented (not given in this paper) and its implementation. However, as suggested in (Yu et al., 1995), the elaboration of this specification for an alternative solution may help to refine the i* models, so that the elaboration of the Albert II specifications for several alternatives could help in the refinement of the i* models containing the rationales for the choice of the new solution.

In order to make this approach effective, we are planning to develop supporting tools allowing the editing of Albert II and i* models together with their storage in a repository (called ConceptBase (Jarke et al., 1995)) where are maintained the traceability links (sketched in Section 3.2) existing between the two descriptions. In another research project, besides the Albert II editor (already available on Windows platforms), other specialised tools are developed for supporting the verification and validation of formal Albert II specifications.

In future work, we would like to extend the presented framework in such a way that the reuse of generic CIM components is encouraged. To this end, we plan to develop a library of com-
ponents (specified in AlbertII) accompanied by information (expressed in i\*) related to the rationales underlying their design.

Acknowledgment
We are grateful to Eric Yu for his comments on the i\* model and the fruitful cooperation established with him within the framework of the European/Canadian ISC-CAN-080 project on Co-operative Information Systems. European Community also supported this research under Project 8319 (Modelage).

REFERENCES

APPENDIX 1 AlbertII SPECIFICATION OF THE CELL SYSTEM

Transp-Controller

DECLARATIONS
STATE COMPONENTS
WS1 instance-of WS
WS2 instance-of WS

Two particular WS among all the WS of the cell.

Busy instance-of BOOLEAN ←→ AGV
Contents set-of PART ←→ AGV
Output set-of PART ←→ WS
Input set-of PART ←→ WS

The controller imports the status (busy or not) and contents of the AGV
and the statuses of the input and output buffers of every WS.

ACTIONS
Transport(PART, WS, WS)

(1) The action of transporting a part from a WS to another.
GetRequest(PART, WS) →→ AGV
PutRequest(PART, WS) →→ AGV

(2) Requests to the AGV to perform respectively a get and put operation
at a particular WS, for a particular part.

LOCAL CONSTRAINTS
EFFECTS OF ACTIONS
OperateAGV: Operating := TRUE

CAPABILITY
F (Transport(p, pos1, pos2) / AGV.Busy)

(3) No transport can occur if the AGV is already busy.
F (Transport(p, pos1, pos2) with pos1 = WS1 / pos2 = WS2)

(4) A transport originating from WS1 can only be destined to WS2.
F (Transport(p, pos1, pos2) with pos2 = WS2 / pos1 = WS1)

(5) A transport destined to WS2 can only originate from WS1.
F (Transport(p, pos1, pos2) / \ p \epsilon pos1.Input)

(6) The transport of a part from a WS to another can not occur if the part
is not present in the output buffer of the origin WS.
F (PutRequest(p, pos2) / Card(pos2.Input) = 0 \lor \ p \epsilon AGV.Contents)

(7) The controller can not ask the AGV to put a part in the input of a WS
if the input is not empty or if the part is not on the AGV.

ACTION COMPOSITION
Transport(p, pos1, pos2) ↔ GetRequest(p, pos1); PutRequest(p, pos2)

(8) Performing a transport consists in asking the AGV to get the part
at the origin and put it at the destination.

COOPERATION CONSTRAINTS
STATE PERCEPTION
K (AGV.Busy / TRUE )
K (AGV.Contents / TRUE )

(9) The status and contents of the AGV are always perceived by the controller.
K (WS1.Output / TRUE)
K (WS2.Input / TRUE)

(10) The contents of the output buffer WS1 and input buffer of WS2 are
always perceived by the controller.

ACTION INFORMATION
K (AGV.GetRequest(p, pos) / TRUE )
K (AGV.PutRequest(p, pos) / TRUE )

(11) Get and put requests are always shown to the AGV.
A Supporting Enterprise Infrastructure Using STEP Technologies

S.P. Reid & Dr. S.K. Banerjee
University of Strathclyde
DMEM, James Weir Building, 75 Montrose Street, Glasgow, G1 1XJ
Telephone: +44 (0) 141 548 2065
Fax: +44 (0) 141 552 0557
E-Mail: s.k.banerjee@strath.ac.uk

Abstract
Many organisations have evolved into disparate functional units through the adoption of a decentralised control structure. Meaningful and effective exchanges of information between these units are required to support organisational integration. This paper proposes the use of STEP technology and software for implementing an integrated information system that can act as a supporting infrastructure to the organisation's functional units. The paper describes a methodology, using a case example, that can be used to develop such an infrastructure.

Keywords
Enterprise modelling, Information Infrastructure, STEP, Express.

1 INTRODUCTION

Manufacturing industries are experiencing a market place dominated by supply, where time based competition is now a reality, and 'time to market' is the challenge for manufacturing management. In order to deal with the ever increasing environmental choices and uncertainties related with manufacturing a more flexible, responsive and creative management is required. Today's competitive organisation is dynamic and constantly changing; unless this change is effectively managed organisations may experience difficulties and future survival. The two main aspects of change that have impacted the organisation are the rapid growth in the number
of products and services, and the speed with which new products and services invade the marketplace. When customer demand patterns are considered in conjunction with this, the management of manufacturing becomes complex and requires a system that is flexible enough to respond quickly to changes and has the appropriate variability handling capability. This flexibility of a system can be provided in the design stage by incorporating the required level of flexibility of resources into the system, and by using rules, procedures and guidelines during the systems operation in order to minimise the effect of change during manufacture (Duimering et al 1992). This implies that to manage today's manufacturing system a more dynamic approach to the design and operation of an organisation is required.

In attempting to become more dynamic in their operation and while trying to improve their overall approach to efficiency, many organisations have adopted decentralised structures and evolved into disparate and autonomous departments or units. However, in trying to optimise the functional efficiency of a unit, one may lose sight of the strategies and goals of the organisation as a whole. Co-ordination of these units therefore is a primary concern in this respect.

Enterprise Integration is a relatively new concept that is trying to address the problems of modern enterprises by integrating different and disparate organisational functions. Enterprise Integration can be defined as the enhancement of the co-ordination between organisations, systems and individuals, that in doing so improve the performance of some of the participating parts, whilst not decreasing that of any. The main goal of enterprise integration is to promote the success of the business itself; its effectiveness being judged by how well the separate enterprise components co-operate with each other to achieve the stated goals of the organisation as a whole. From this point of view it can be seen that there is a need for effective communication through an integrating infrastructure that would link the disparate departments.

It is possible to develop an integrating infrastructure in a particular area of interest to aid systems managers by using what is available or known at present. Such an infrastructure could be created using a neutral data format so that it will enhance the process of information management and transfer between the participating units of a system. Having such an infrastructure would increase the availability and ease of flow of appropriate information between the departments even with different computer applications running on differing hardware platforms.

2 MODELLING THE ENTERPRISE

An essential part of the integration process is the creation of a valid enterprise model. A comprehensive review of issues regarding enterprise modelling can be found in Patankar and Adiga (1995). An enterprise model will represent the components of an organisation, their function, their interactions and the information flow between them. This enterprise model can then be used as the basis for the creation of the integrating infrastructure. There have been three methods suggested for creating enterprise models and these are (Petrie 1992);

- The Master Model approach,
- Unification, and
- The Federated approach.
The authors have adopted the master model approach; this approach rests on creating a single reference model from which it is possible to derive and extract sub-models that represent the required views of the organisation from different perspectives. This principle can be seen in Figure 1. The IEM concept (Mertins et al 1992) also follows this approach.

In order to create an enterprise infrastructure that will bring together disparate organisational functions it is necessary to create accurate models that can be used to represent the different functional areas of the organisation. Such functional models can be created using a neutral information modelling language, such as Express and then can be linked, through their commonalities, to form a larger 'Master Model'; a methodology also advocated by Hars and Scheer (1992). It can then be used to create valid instances of the companies' requirements.

The developed master model may be used as the basis for defining the structure of an object oriented database implementation that will hold the organisations data in a convenient form and location, and from which users can access the information they require.

However, before this can be done the important functional aspects and entities within the organisation have to be identified so that nothing of importance is left out of the models. The process of identifying these important organisation qualities is covered during the requirement's analysis phase. The determination of the requirements of the organisation is an integral part of the modelling process as it involves creating a set of requirements that fully represent the company's needs and each unit's contribution to them without overlooking any particular aspect of the company. It is therefore imperative that there be a high level of communication between developers and management during the requirement's analysis phase.

The requirements, and hence the models, are developed through discussing the problem domain with the people who are most familiar with it, e.g. those responsible for activities and information, as well as the users of the information (van Griethuysen 1992). Methodologies such as Quality Function Deployment (QFD) could be used to create such a set of requirements.
3 ENTERPRISE MODELLING USING EXPRESS

Having completed the requirements phase the Express information modelling language (Schenck and Wilson 1994) can be used for creating models of an organisation's information system. This modelling language is now an international standard that has been developed as part of the STEP\(^1\) standard and its formal structure is defined in (ISO 1991). Express was originally designed for the purpose of creating models that could be used to facilitate the exchange of data concerning mechanical products. The scope of its use has broadened considerably since.

The Product Data Exchange Specification (PDES) provides a way of representing product data in a neutral format throughout the products life cycle. PDES has three important components (Qiao et al 1993):

- Reference Models,
- The Express modelling language,
- The STEP file structure.

The development work within the STEP domain has been conducted within a variety of areas. These development areas are commonly referred to as 'Parts' within STEP, the number of which is continually growing. Each of these parts contains information relating to a particular aspect of the standards for product representation and description and usually incorporate reference models that have been developed using Express.

The whole basis of STEP is, as its name states, to define the standard for the exchange of product model data, and it is with this in mind that most of the available STEP related tools have been developed. The authors believe, however, that it is reasonable to use these technologies for the purpose of creating an enterprise infrastructure that will encompass and support the needs of the whole organisation.

The Express information modelling language was chosen for use in the model development phase for the following reasons:

- It is now an international standard information modelling language, defined in ISO 10303, and has a large and growing user base.
- It is computer interpretable, and tools are available that can be used to validate and manipulate Express models. There are tools available, called Express Parser's, which can be used to check an Express models' semantic and syntactic correctness. Examples of these include Fedex and ICE methods, developed by NIST\(^2\) and The CadLab at The University of Paderborn.
- There is a wide and increasing variety of tools available in the market place as well as in the public domain that can be used for the development of Express information models and the data instances for them. A comprehensive list of Express tools and services can be found in (Wilson 1995).
- It uses a neutral data storage format, i.e, the STEP file format, which greatly enhances the ability to exchange data between applications readily. The ISO 10303-21 (ISOb 1991) constrains the number of instances within a single STEP file to \(10^9\) instances (Lehrenfeld et al 1994).
- It has a graphical subsection in its definition that allows for models to be developed using graphical tools as opposed to just text editors. This greatly increases the ability of

---

1 STEP is the Standard for the Exchange of Product model data.
2 NIST is The National Institute for Standards and Technology.
developers to visualise a model and its inter-relations as well as to edit and update Express models.

Following the Master model approach, the use of Express brings some distinct advantages to the modelling process. It is possible for the master model to evolve over time so as to encompass the whole of the organisation. This makes it possible for the model development process to be conducted in an incremental manner, if required, where organisational units or departments are modelled and then incorporated into the Master model.

The use of a single model format facilitates the integration process as new models can make use of the developed model and can therefore incorporate any commonalities that already exist in the model. This has the effect of reducing the problem associated with semantic integration as there will only be one representation of each requirement within the model. The viewpoint for a particular department may require information concerning an entity that is not required by any other departments. This information can be masked from other viewpoints by not including the attributes in the extracted model. This does not mean that the information will not be there; it will just not be visible to those who do not require that particular information.

In order to overcome consistency problems between model formats, Felser and Mueller (1994) propose an extension to the Express modelling language that allows the modelling of processes. The proposed extensions include;
- interface declarations,
- local functions and procedures,
- system and process declarations, which are all placed within the existing entity declaration. While this extension will not conflict with the current Express specification (ISO 10303-11), there will be problems when trying to validate these models with existing STEP tools.

4 DEVELOPING THE INFRASTRUCTURE

The process of creating a supporting infrastructure, as shown in Figure 2, starts with the organisation conducting a requirement's definition exercise that will yield the data and information flows that are critical to the organisations functioning. This will also involve identifying information that is needed by specific users and departments within the system (functional units). The requirements generated can then be used to create an Express model of the organisations functional requirements. This can be done by making use of an Express-G graphics package. This will allow the user to create and link entities, which will represent the real world objects that are of interest, to form an enterprise model. By using the Express-G package the model developers will be able to edit with ease and update the enterprise model to incorporate all of the required information. The Express-G package generates the Express code corresponding to the developed enterprise model. This can then be validated by an Express Parser that will check the model against the definitions and constraints defined within the Express modelling language (ISO 10303-11). If there are any errors in the model they will be identified and the model can be modified to erase them.

When the developers are happy that they have a model that accurately represents their system and includes all the information that may be required by the system users, the enterprise model can become the master model. Preliminary work has been conducted into creating a supporting infrastructure that utilises the NIST Data Probe software for creating and editing instances of the manufacturing system (Reid and Banerjee 1995). This system use importing
Figure 2 Outline of the process of creating a supporting infrastructure.

Figure 3 Structural overview of the production planning and control system.

Figure 4 Entity relationship diagram with attributes for Cell Co-ordinator data model.
and exporting STEP files as its means for updating the information in different viewpoints of the manufacturing system.

A model of a manufacturing planning and control system was developed for a manufacturing company (Zhou 1993), the structure of which is shown in Figure 3. This system was originally defined using entity relationship diagrams, as in Figure 4 and data flow diagrams, which were subsequently translated into Express; an example is shown in Figure 5.

Although the arrangement developed through the preliminary work for editing and updating the manufacturing system information may be adequate for some implementations, an implementation that made use of the STEP Standard Data Access Interface (SDAI3) and a database system would increase the usefulness and applicability of the infrastructure. To this end the master model can be used to create an implementation of an object oriented database. This is done by modifying an express parser so that it accepts an express model, in this case the master model, and outputs the implemented database structure which is a mapping from the master model into the database system (Luhrsen and Krebs 1995). The modifications required to the express parser depend upon the particular database system being used. This is commonly known as creating a 'back-end' to the parser. The structure of the Express modelling language lends itself well to being translated into databases that are C++ based as they have similar data structures.

The next stage in the development process is to create an implementation of the STEP SDAI. The SDAI is used to control the access to information relating to specified express models. Implementations of the SDAI may involve the manipulation of STEP files and their data or the manipulation of a database and its information. In the case described the structure of the database is defined by the structure of the master model. It can be seen in Figure 5 that if

---

Figure 5 Entity level diagram in Express-G, of the Assembly schema, for use with the Data Probe software.

---

3 The SDAI is defined in STEP Part 22, and a C++ binding to the SDAI is defined in Part 23.
a user's information requirements are defined by a sub-model of the master model, then the information held within the database relevant to the end-user will correspond to the same sub-model. Therefore the SDAI will provide the end-user access to the information based on their specific sub-model. The actual data that the end-user will be able to edit and update will be defined during the database user set-up procedures. The ownership and access of data are therefore predefined.

In order to be able to provide the required information to the end-users, when they need it, the SDAI must have some means of communicating with the database. The SDAI does this by having function calls to the database systems manager (DBMS) incorporated into its definition (Goh et al 1994, Rando and McCabe 1994). Therefore, if the end-user wants to query the database they initiate the query at the SDAI interface, which then calls the database management system. The database management system interrogates the database and returns the required information to the user. This process can be seen in Figure 6. Because the implementation details of the SDAI have not been defined it is possible for all SDAI implementations to have the same definition and user interfaces, no matter what database system they are implemented to.

With the supporting infrastructure developed and the SDAI implemented it is possible for company specific programs to be written that will provide the end-user with the information they require when they need it. These programs will be extensions to the SDAI user interface, in that they will be connected to the SDAI, but the end-users' display will show the information that is needed in the required format, accessed via the SDAI. In this fashion the information obtained by the SDAI is manipulated into a form that is most suitable for the specific end-user.

Figure 6 Query initiation and response process using an SDAI implementation together with an object-oriented database and management system.
5 CONCLUSIONS

In today's dynamic working environment it is necessary for organisations to have a supporting information infrastructure that will reduce information duplication and redundancy, while providing the end-user the information they require. A reconfigureable infrastructure like this can be created using current STEP technologies. The research undertaken by the authors has shown that it is possible to use these current tools for the creation of a supporting information infrastructure that allows updating of information from different enterprise viewpoints. It can also be noted that the introductory period for gaining knowledge on the use of these tools is very short. The authors are currently investigating the use of a SDAI implementation with an object-oriented database as a natural progression in this area of research.

While there is no scope for modelling processes within the current ISO definition of Express we believe that the modelling of processes and change should be tackled in another manner. While the information within a system may change its state there is no mechanism within Express to conduct this change automatically. Therefore, to be able to create a supporting infrastructure that can adapt to these changes it is necessary to develop procedures that can be used to change the state of instances within a database. The authors are investigating the use of object-oriented methods, such as Booch (1994), for providing the facilities for the definition of programs which will then be able to manipulate the stored data. These programs will be developed from the information requirements of the end-users and will allow the information to be presented in a usable form when the users require it.

Ellis et al (1994) proposes a similar approach for describing a reference model for open distributed processing (RM-ODP), where the information view is described using IDEF0 and Express, and a computation view is described using Booch diagrams.

Using the methodology outlined in this paper it is possible to create a supporting infrastructure for an organisation that can be extended to include more aspects of the organisation as and when they are modelled. The use of STEP tools allows for the rapid reconfiguration of viewpoints, or sub-models, of the Master model of the organisation.

6 REFERENCES


Rando, T., Mccabe, L., 1994, Issues in implementing the C++ binding to the SDAI. Computer Standards & Interfaces, 16(331-340).


Wilson, P. R., 1995, Express Tools and Services. NIST, Building 220, Room A127, Gaithersburg, MD 20899.


7 BIOGRAPHY

S.P.Reid is a PhD research student at the University of Strathclyde whose background is in Mechanical Engineering, and his current research work is in STEP technologies for manufacturing support system development.

Dr.S.K.Banerjee is a senior lecturer at the University of Strathclyde and is a member of the task force on Enterprise Integration, IFAC/IFIP WG 5.12.
On reference modelling of integrated manufacturing systems using OOA

G. Vosniakos* and M. Williams**

* National Technical University of Athens, Dpt of Mech. Engng, Patission 42, GR-10686, Athens, Greece, tel +301 7723691, fax +301 7723689, e-mail Vosniakos@central.ntua.gr

** UMIST, Dpt of Mechanical Engineering, PO Box 88, Manchester M60 1QD, Britain, tel +44 161 2003800, fax +44 161 2003803, e-mail mcjicmaw@fs1.me.umist.ac.uk

Abstract

This work advocates the use of Object Orientation as basis of a methodology for creating reference models of manufacturing systems. Work on reference modelling of manufacturing systems integration is reviewed first. Then this work's own standpoint on reference modelling is outlined with definition of the main concepts. Reference models are distinguished from actual-implementation models through the need for generalisation, abstraction and expansion ability. Coad and Yourdon's Object Oriented Analysis is used to demonstrate reference modelling of structural, transformation and procedural aspects of a batch-processing shop-floor system, explaining how manufacturing systems entities are mapped to OOA entities. The main concepts introduced along with discussion of relevant examples are a phenomenological View, application-related Views, abstraction hierarchy of objects and expansion ability of the model to advance from abstract to environment-specific reference models. Modular instancing of entities is also discussed in the context of further work.

Keywords

System Analysis, Manufacturing Systems, Shop-floor, Coad-Yourdon’s OOA.

1 INTRODUCTION

Manufacturing system integration typically addresses many subsystems performing complex sequences of actions, which possibly run concurrently. Cross-area integration in manufacturing systems refers to the specification of common goals, outputs from one area forming inputs to other areas, cross-area consultation between functions, and repercussions of parameter changes in one area onto other areas (Burbidge, 1987).
In an effort to design and implement integrated manufacturing systems the concept of reference models has been explicitly or implicitly recognised to be of fundamental importance (Biemans, 1991), (Jochem, 1989). In general, the term 'reference model' describes an 'ideal' or 'target' configuration of a manufacturing system or of parts of it and therefore it has a 'prescription' or 'recipe' connotation. A reference architecture by contrast defines a modelling methodology (rules referring both to the modelled domain and to the modelling task and structures, possibly including a modelling language) (Aguiar, 1995). Another term which is often used in literature as equivalent to a reference model of a high level of abstraction is 'generic model', e.g. in a production control context (Filip, 1993).

According to one classification, there are four main types of use of reference models for manufacturing systems: Benchmarking, Diagnosis, Design, Evaluation (Bohms, 1993).

A CIM Reference model in the context of process industries is given in (McCarthy, 1990). It consists of a number of sub-models, namely organisational, control strategy, information flow and network connectivity used to derive the functions needed in a plant information system.

In designing automated production control systems, system aspects captured by a reference model are categorised as behavioural, information and functional (Lhoste, 1993). A 'diachronic' system life cycle is advocated, using the equivalent of 'invariant' libraries of standard and re-usable behaviour, information and functions.

Taxonomies form a good baseline for the development of reference models, one obvious thing they lack being functional interactions of the categorised objects. A taxonomy example referring to materials handling environments (Krni, 1995) can be easily paralleled to an 'uninstantiated' reference model. Another six-level taxonomy of typical CIM activities is the basis for planning the Information Systems in CIM (Camarinha-Matos, 1994).

Research needs with respect to CIM architectures defined in (Aguiar, 1995) follow four streams: extension of CIM-OSA, development of object-oriented modelling tools, development of standard infrastructural type of tools and building information modelling and manipulation tools. The need is recognised for populating a system engineering workbench with a library of reference models that encapsulate knowledge of user requirements associated with resource capability and of information technology solutions based on enterprise domain typification.

2 OBJECT ORIENTED ANALYSIS

The methodology chosen in this work to create reference models of (integrated) manufacturing systems is Object Oriented Analysis (OOA) by Coad and Yourdon (Coad, 1991). An OOA model can be derived in five separate modelling steps: finding Classes and Objects, identifying Structures (whole-part, gen-spec, instance connections and event-remembered objects), identifying Subjects, defining Attributes, defining Services.

Manufacturing system reference models need to reflect the Structural, Transformation and Procedural definition of systems. Structural definition is the description of all physical and conceptual elements of the system. Transformation definition is the description of the functions performed by the system elements. Procedural definition is the description of time-phased interactions of system elements, i.e. the dynamic behaviour of the system (Hitomi, 1990).

The transformation definition of the system in OOA is given through services indicating what behaviour will be provided by an object within a class. The procedural definition of the system, i.e. communication and sequencing requirements, in OOA involves defining communication messages between objects. These messages invoke a response such as replying to the original object, displaying results, monitoring a dynamic attribute etc. (Coad, 1991).

Before demonstrating the adequacy of OOA for creating reference models of manufacturing systems it is appropriate to highlight the distinguishing characteristics of reference models, as these stem from the
requirements for generalisation, abstraction and expansion ability.

3 REFERENCE MODEL CONCEPTS

Generalisation ability is required in order to cover as many possible implementations of the proposed reference model as possible. Abstraction ability is required in order to modularise the model according to a variety of changing criteria. Expansion ability is needed in order to move from an abstract to a more concrete and finally to an implementation version of the system. Note that the term ‘implementation’ of a reference model is used as synonymous to realisation of a physical or software system corresponding to that model.

Views
In this work, a reference model is considered to support one or more ‘Views’ each of which is tied to a particular application. This is rather similar to the concept of external schemata in a database model. A phenomenological view is considered to form the basic reference model of any manufacturing system. This view describes how the system appears to be working to an external observer. Typically, its objects, attributes and services are generic enough to be applicable to a variety of circumstances. Therefore this view cannot be concerned with details of complex algorithms such as planning and scheduling. Any other view of the manufacturing system essentially extends or abstracts certain areas of the phenomenological view.

Views may share objects and some of their attributes, services, messages and instance connections, in which case these objects, attributes etc. are termed common. Figure 1 illustrates this notion. Object_1 is common to views P and I, whereas Object_2 is common to views P, I and 2. Instance connection ic_1 belongs to view P, which means that it refers to aspects (attributes and services) of Objects 1 and 2 that pertain to View P. Message connection mc_3 refers to two services pertaining to view I, hence it belongs to view I, too. Views may also possess objects and some of their attributes, services etc. exclusively, in which case these objects, attributes etc. are termed private.

Abstraction hierarchy
For any specific View (application) a hierarchy of reference models may be defined, starting at the top with a model generic enough to be both configurable to individual company environments and reusable in different situations. Subsequent levels in the hierarchy, corresponding to different levels of abstraction, would add progressively more detail to the top level model, leading towards specific implementations. This is possible according to the principle of inheritance where the more specific class inherits the characteristics of the more generic ones. In addition, relationships can be defined between classes that belong to the same level of abstraction, thus becoming part of a model extension, see Figure 2.
In order to carry out an expansion of the model a set of transformation rules is needed referring to classes, attributes, services, instance connections and message connections. Model expansion takes two forms: adding classes / objects to a new level in a generalisation-specialisation structure and adding new classes independently of any existing structure.

In the first case the class / object inherits attributes and services of the level above. Any new attributes and services will be simply added. Instance connections and message connections will also be added to reflect the behaviour of the new class / object.

In the second case the new object will in most cases define in further detail some aspect of an existing object, hence the initial object can be considered to be split into at least two new objects. These will be related by instance connections and possibly by message connections, too. The instance and message connections of the initial object will have to be preserved in meaning, but will probably change in form, see Figure 2. If a service of the original object which is involved in a message connection is retained in the new picture, then the message connection is most probably to be retained, too. If, on the other hand, the service is transformed into separate services, these will probably have to communicate both with each other and directly or indirectly with the objects to which the initial message connection referred. The above depend very much on the nature of services, of the instance connections and of the message connections, hence they are not cast in stone.

4 A SHOP-FLOOR REFERENCE MODEL FOR BATCH PRODUCTION

Due to the size even of a part of a manufacturing system and the many possibilities for different applications pertaining to any such sub-system, only the phenomenological view has been concentrated on in this work. The sub-system modelled was the ‘production execution’ on the batch-processing shop floor for mechanical components, mainly because of its relevance of the authors background. In a later expansion phase additional system responsibilities can be added. The main features of the reference model developed follows.

Subjects, classes, objects and attributes
Four subjects were identified: Shop-floor, Product, Inventory and Personnel, see Figure 3. The shop-
Fig. 3(a) Reference model for subject Shop floor
Fig. 3(b) Reference model for subject Product
On reference modelling using OOA

Fig. 3  (c) Reference model for subject Inventory
(d) Reference model for subject Personnel
(e) Subject connections - see also Appendix
Table 1  Instance and message connections across subjects - supplement to Figure 3(e)

<table>
<thead>
<tr>
<th>Index</th>
<th></th>
<th>From - to</th>
<th>Index</th>
<th></th>
<th>From - to</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.1</td>
<td>ic</td>
<td>AccessStatus-Hardware</td>
<td>1.4.1</td>
<td>ic</td>
<td>Employee-Shop Floor (part of)</td>
</tr>
<tr>
<td>1.2.2</td>
<td>ic</td>
<td>NcPartProg-DownLoad</td>
<td>1.4.2</td>
<td>ic</td>
<td>Employee - Hardware</td>
</tr>
<tr>
<td>1.2.3</td>
<td>mc</td>
<td>AccessStatus-Hardware</td>
<td>1.3.1</td>
<td>ic</td>
<td>CutTool - ToolMag</td>
</tr>
<tr>
<td>1.2.4</td>
<td>mc</td>
<td>Hardware-AccessStatus</td>
<td>1.3.2</td>
<td>ic</td>
<td>CutTools - ToolHolder</td>
</tr>
<tr>
<td>1.3.3</td>
<td>ic</td>
<td>MeasureTools - ToolMag</td>
<td>1.3.4</td>
<td>ic</td>
<td>FixtureTool - McToolHardware</td>
</tr>
<tr>
<td>1.3.5</td>
<td>ic</td>
<td>Inventory-MtlHandleHardware</td>
<td>1.3.6</td>
<td>ic</td>
<td>Requisition - Employee</td>
</tr>
</tbody>
</table>

floor subject includes a large Gen-Spec structure called Hardware, see Fig 3(a). All its instances have a location, description and identity inherited from the Hardware class. Machine tools are described by their capabilities, maintenance related data and Group Technology coded operations available. As a further specialisation there are NC, conventional and material handling machines. They possess, among other attributes, status, capability specifications and maintenance data.

In the Product subject, see Figure 3(b), the class Product has customer, works order number, due date and test procedure attributes, the latter emphasising the necessity for quality. A product is made up of parts manufactured in house (McBatchPart), purchased from outside (PurchasePart) and subcontracted (SubconPart). McBatchPart has a priority attribute useful for scheduling, a DefectStatus attribute used in Quality Assurance and a current status. Subcontracted parts have a certification attribute emphasising Total Quality. There is also provision for parts partly manufactured in house and partly subcontracted on a per operation basis (McSubconPart). This is connected to SubconPart and McBatchPart in a lattice structure.

Still within the Product subject, AssemblyDrawing is an object representing product structure. It consists of PartsList and of DetailDrawings. There are also objects representing the product structure, namely: PartsList with part name, reference number, part number, part data and number off attributes, and DetailDrawing with drawing number, manufacturing data (tolerances, dimensions, surface finish etc.) and Group Technology codes. There is also a ProcessPlan object used on the shop floor for reference, whose attributes define sequence of operations and machines necessary for production. In addition, a Schedule object is needed for low-level scheduling, with main attributes priority and due date.

Inventory represents all material resources required by a production system, such as raw materials, general consumables, and tools, see Figure 3(c). Each instance of the inventory class has location, reorder policy, inventory_status (standby, monitor, order), number_in_stock, supplier and cost attributes. RawMaterial and GeneralInventory classes possess Certification attributes. Tools inventory is broken down into cutting tools, fixtures and measuring tools. Cutting tools have a tool life, a classification code detailing dimensions, operations etc. and company reference number, e.g. ISO. Fixtures need a drawing, too, whilst measuring tools are different in that they have a measurement scope, and they need calibration which is carried out according to a standard.

Personnel is the smallest of the subjects, but the corresponding external mapping is detailed, reflecting the importance of human involvement in a manufacturing system, see Figure 3 (d). The Employee attributes include, among other attributes, authorisation level for accessing data and training qualifications.

Instance Connections

There are several instance connections between objects of the model. A detail drawing, for example, is...
associated with a part machined or subcontracted and also with a process plan (one-to-one relationship),
It is also associated with the corresponding parts list and perhaps with an NC program. Note that these
are all associations within the same subject.

As an example of instance connections spanning more than one subject refer to the McBatchPart
object. A connection occurs with the ProcessPlan object, as each part has to have a specific process
plan produced before it can be manufactured (one-to-one relationship). Additional connections occur
with objects AssemDrwg and DetailDrwg, mapping association with the appropriate engineering
drawings. Connections also occur with Employee and Hardware. These connections map the
association employees and equipment have with a part as it is processed and reflect many-to-many type
of relationships, because specific employees and pieces of machinery can process many different
manufactured parts and vice versa.

Event-Remembered objects include:
• AssignJob, which relates a schedule and an employee and needs machine identification and date-
time attributes
• TestProduct, which relates Product and Employee and has a date-time attribute
• AccessStatus, which relates McBatchPart, Employee and Hardware at a generic level representing
the system’s capability of retrieving current, up-to-date status information stored in relevant
dynamic attributes. This is broken down further into StatusProgress referring to machined batches
and StatusHard referring to hardware status.
• Requisition relating Inventory and Employee to model the allocation and withdrawal of inventory
items within the manufacturing system. Attributes are: DateTime, OrderNo, Amount and Location.
• Download relating NCPartProg and Controller / HanController objects, attributes being mainly
time and date.

Services and Communication
Services related to Product objects can be noticed in the class boxes in Figure 3(a)-(c), e.g. :
• Assemble Product, inspect TestProduct, maintain McToolHard, setup NcMcTool, loadProg
Controller : typical execution type of services.
• Check capacity : attached to AssignJob object it represents an algorithmic check of availability of
shop-floor capacity before assigning a job.
• CalculateStockLevel Inventory : typical calculation type of service.

Message connections are illustrated by referring to McBatchPart, see Figure 3. Processing progress is
recorded within the object by the different states of the dynamic attribute CurrentStatus. This needs to
be accessed, access being initiated from within the object StatusProgress by the service CheckProgress.
The service CheckStatus is then activated within the object AccessStatus which sends a message to
McBatchPart activating the service CalcProgress. This service calculates the current position /
operation of the batch within the system and sends the result back to AccessStatus via the message
connection. The result is then displayed, recorded or printed out using the service DisplayStatus.

5 DISCUSSION

In a manufacturing systems context Object Orientation has the potential to model data, knowledge,
hardware, information, communication and events as separate objects. In effect these constitute the
structural, transformation and procedural definition of a manufacturing system. In addition, objects
which are found in one manufacturing system are very likely to occur in other Manufacturing systems
with little or no modification. These objects and their relevant structures can be directly re-used,
making it ideal to achieve the level of abstraction necessary for the production of reference models.
Part Two  Modelling of Products, Processes and Systems

Note that no subject of the model can be completely self-contained. There will always be instance connections and messages with objects belonging to other subjects. Sometimes it is expected that for some of the classes it will not be clear to which subject they belong.

Time sequences of events are not conveniently shown, certainly as far as visual interpretation of the model is concerned. This is an inherent weakness of OOA methodology and could be cured by either adopting a complementary methodology such as Activity Cycle Diagrams or by suitably modifying OOA, e.g. through definition of special types of services and special notation for message connections implementing event sequences.

Coad-Yourdon’s OOA has already been proposed as a modelling methodology for manufacturing systems, albeit not from a reference standpoint. However, it was slightly modified and unnecessarily complicated into a ‘new’ methodology named HOOMA (Wu, 1995). The differences are first that HOOMA starts with a functional decomposition of the system in order to define in essence modelling domains. Then, functional decomposition in the form of Subsystem Relationship diagrams is carried out further before class and object definition. Objects and classes are distinguished into clients and servers, the distinction having though no apparent subsequent use. In HOOMA’s whole-part structures parent objects cannot exist at the same time as their parts (children). This restriction seems to wipe out any local knowledge of system structure. State Transition Charts and Activity Cycle Diagrams are used mainly for improving ease of derivation of the correct instance and message connections.

As far as a comparison of OOA with CIM-OSA is concerned, it was noted (Nager, 1995) that CIM-OSA leads to complicated lengthy models, compared to SADT and to Rumbaugh’s Object Modelling Technique. 3 X 3 X 4 types of CIM-OSA models are potentially needed for comprehensive representation of a manufacturing scenario. However, it must be recognised that several of these types are not straightforward to represent using other methodologies. In brief, it can be stated that CIM-OSA is just a modelling framework providing tools and methodology for modelling an enterprise or parts of it. Instanced models are not available and it is up to the users to develop them by plugging relevant knowledge into the modelling shells, as was done, for instance with respect to a CAD / PPS interface applicable to electromechanical product manufacturers (Nager, 1995).

The phenomenological view of the shop-floor model is merely a description of what should happen in a ‘typical batch processing shop floor’. As such it could be termed an ‘execution’ View. There will certainly be applications referring to the Shop Floor with a different flavour, e.g. design, performance evaluation etc.

Each of these applications will make use of the objects, structures etc. contained in the phenomenological View of the model, but will need to extend it by adding further elements (objects, instance connections, messages, services etc.). Thus, the phenomenological View is also the ‘core’ View around which further extensions can be built to construct further Views. It should be noted that the phenomenological View needs to be generic enough to allow any additional View to be built on top of it. A preliminary discussion of such Views follows.

Specification of Shop Floor control: This View requires the existence of a controller hierarchy in the model for short term planning and monitoring of execution of production plans. Control components would need to be specified in terms of their function, status, and interaction. These would communicate with dynamic attributes of specific objects of the ‘core’ View, for instance via event-remembered type of objects.

Facility layout planning: This View would require adding e.g. location co-ordinates and perhaps location preferences / constraints to Hardware as well as demand indicators on top of the existing group technology codes to McBatchPart. The decision scheme (minimum cost equation, rules etc.) could be encapsulated into the additional object Layout.

Specification of machines and computer systems: Physical implementation can be achieved in many ways, requiring breaking down abstract objects into a set of physically implementable ones. The corresponding View would then be a ‘concretisation’ of the abstract ‘core’ View.
Specification of interfaces with other manufacturing system areas: A number of reference models pertaining to individual manufacturing system areas may be constructed. These are expected to use many common objects, though not with a common set of attributes, providing a basis for integration. Integration should be enhanced by specifying explicit links through interface objects.

To illustrate the notion of collaborating views, an example of the possible role of the McToolHardware object in four views of a manufacturing system model is considered. The phenomenological view includes an attribute (Location) of the Machine object describing its location in a 3D space grid. In a layout design view a procedure (Optimise_location) is required to calculate the optimum location of the machine. In a physical element specification view instancing of the Machine object will depend on the technological characteristics derived after observation of all available instances of a Process_Plan class, hence there is a connection (both instance and message) to this object. In an interface specification view a subset of attributes of the Machine object can be identified for communication with other subjects, e.g. Production_Planning. For instance the Location attribute may be required by a service (Assign_LeadTime) calculating transportation time as part of the production lead time. This service may be attached to a Schedule object and effect its LeadTime attribute.

The model presented in Figure 3 can be seen as a generic model at the highest level of abstraction. There are, however, certain classes that have been expanded into more specific classes, such as the ToolInventory class. The CutTool subclass needed then to be related to ToolMag (and ToolHolder) classes with new instance connections which could not be applied to the parent class Inventory, covering diverse inventory items. It could, though, be related at the other end to class NcMcToolHardware because of its special one-to-one whole-part relationship with class ToolMag. These ‘expansion’ classes could have been omitted in the first place, with loss of all relevant attributes and relationships, and added at a later stage, observing the transformation rules mentioned above.

Further work is needed towards a structured abstraction of a manufacturing system by grouping objects into pre-defined generic classes such as: Information, Processors, Planners, Controllers, Calculators, Materials, Interfaces. In addition, the shop-floor model presented above needs further validation in order to become a true reference model, since this was based on own experience and academic material. Given the size of this task, it is no coincidence that no standardisation of any kind exists in the domain of batch manufacturing industries for mechanical engineering components.

6 CONCLUDING REMARKS

This work explores concepts relating to building reference models for integrated manufacturing systems. Object Orientation is advocated as a suitable paradigm for expressing such models and the application of Coad and Yourdon’s Object Oriented Analysis is shown to be an adequate tool for this task. In particular, it is advocated that the OOA method in its pure form is better than any method based on it with a few ‘enhancements’ and modifications, since apart from other reasons, it retains the possibility for creating ‘executable’ models, i.e. a direct channel to software development based on the model.

The Shop-Floor model constructed using OOA can be seen as a ‘core’ model which can be used as basis for building application-specific Views relating to the same domain. Categorising the types of entities encountered in this domain, i.e. raising the level of abstraction, should provide for a modular approach to building ‘custom’ Views. Further work is being carried out in this direction.

ACKNOWLEDGMENT

Mark Williams acknowledges the financial support of the Engineering and Physical Sciences Research Council of Britain.
7 REFERENCES


8 AUTHOR BIOGRAPHIES

George Vosniakos graduated from the National Technical University of Athens in 1986. He obtained MSc and PhD degrees from UMIST in 1987 and 1991 respectively for work on CAE. He then worked as a Research Manager for a German CIM software house and in 1993 he joined UMIST as a lecturer in Manufacturing Systems. In 1995 he was elected Assistant Professor in Manufacturing at NTU Athens. He is the author of 22 articles on CAE topics.

Mark Williams graduated from Trent Polytechnic in 1992, having worked for GEC - Turbine Division for about 3 years. He then, obtained an MSc in Advanced Manufacturing Technology and Systems Management at UMIST. Since 1994 he has been carrying out work on Business Process Re-engineering towards a PhD degree at UMIST.
Metamodels for Information System Modeling in Production Environments

G.S. Mannarino*, G.P. Henning+ and H.P. Leone*.
* Instituto de Desarrollo y Diseño (INGAR), Fundación ARCIEN - CONICET, GIPSI-FRSF-Univ. Tecnológica Nacional
Avellaneda 3657, 3000 Santa Fe, Argentina, Tel.: 54 42 553439, Fax: 54 42 553439, E-mail: hleone@arcride.edu.ar
+ Instituto de Desarrollo Tecnológico para la Industria Química (INTEC), CONICET - UNL
Güemes 3450, 3000 Santa Fe, Argentina, Tel.: 54 42 559175, Fax: 54 42 550944, E-mail: ghenning@intec.unl.edu.ar

Abstract
Production environment models can assist the goal of enterprise integration by helping to represent, and later, analyze and evaluate the structure of the enterprise activities and the interactions among them. This paper presents advances in the development of an open model architecture consisting of three layers: (i) Metamodel layer, (ii) Model layer and (iii) Universe of Discourse layer. The contribution focuses on the introduction of task and domain metamodels to be used as templates in the construction of a production environment model library. In order to develop such models the Object Technology, enriched with the concepts of version, perspective and evolution was adopted.

Keywords

1 INTRODUCTION
Modeling is an invaluable approach to addressing the Enterprise Integration (EI) Problem. Several important modeling methodologies, from different application domains, have been
proposed in this field (Vernadat, 1992; Berio, 1995). The first International Conference on Enterprise Integration Modeling Technology (ICEMIT) roughly identified the three following types of approaches to the problem of syntactic and semantic model integration: Master, Unified and Federated Models (Petrie, 1992). This paper presents advances in the development of a set of metamodels that belongs to the first group. The proposed metamodels allow the description of a variety of production facilities, acting as a foundation to the analysis and design of an integrated information system.

This work focuses on the introduction of a set of task and domain metamodels, based on the Object Technology (Stefik, 1985; Kiczales et al., 1991; Booch, 1994; Jacobson, 1994), to be used as templates in the construction of models of production environments. The primary reason to develop such models is to supply a mechanism for understanding, analyzing and evaluating the production enterprise.

The aim of task metamodels is to describe an organization in terms of (i) the activities that are to be performed so as to achieve the organization purposes such as Production Planning, Product and Process Engineering, Marketing, Distribution, Business Management, etc., (ii) the physical and information resources that are utilized by each task performed in the organization and (iii) the role that each resource plays in relation to a task (whether it can be consumed, modified, used exclusively by the task, etc.)

Domain models focus on the identification of the entities that represent an organization, their relationships, and properties. For example, they will classify batch plants in terms of their operating strategies (i.e., multiproduct, multipurpose and multiplant), an equipment item in terms of its functionality, a production campaign in terms of its starting and finishing times, the product it manufactures, the equipment unit it uses, etc.

2 ANALYSIS OF THE PROPOSED METAMODELS

Integrated production facilities should properly administrate a huge amount of information having a very complex structure and a variety of formats, such as text, graphics, constraints, engineering procedures, etc. (Vernadat, 1988; Meyer, 1990). Therefore, the requirement exists for an appropriate representation mechanism able to express all sorts of enterprise knowledge. The Object Oriented Paradigm, enriched with the concepts of version, perspective and evolution (Mattsson, 1988; Stephanopoulos et al., 1990; Ziegler, 1984; Li and McLeod, 1994; Stefik, 1985), was adopted to develop the proposed representation.

The task and domain object metamodels presented in this paper are first introduced by means of a pictorial notation that is a modified version of the one proposed by Martin and Odell (1992). However, models are later expressed in terms of a formal grammar that defines the modeling language primitives (i.e., objects and relationships linking them) as well as cardinality constraints that apply on relationships. Having such a grammar can help in the verification of the models obtained for a specific production environment when applying the metamodels that are proposed in this contribution.

The suggested model architecture is depicted in Figure 1. This figure, showing three different layers of models, indicates that each layer depends on the primitives provided by the previous layer. The bottom level represents the Metamodel layer. At this layer, generic constructions that pertain to production environments and are independent of any application, are introduced. Classes defined at this layer encapsulate the fundamental model concepts and,
more importantly, the production environment underlying semantics, e.g., the notion of function, task, resource, etc.

In the next level, associated with the Model layer, the characteristics of the application domain are abstracted. This layer reflects, through different models, the operating procedures, tasks and resources associated to industrial enterprises. The objective is to develop an open model library, that could be specialized to reflect the peculiarities of different types of production facilities. This library contains, among others, the following models: organizational structure, functional structure, tasks involved in the different enterprise activities, etc. When these models are specialized, new models belonging to the same level are generated. However, this layer is also “open” to the generation of new models that could be developed from the instantiation of metalevel primitives. Finally, the top level, called Universe of discourse layer, contains the real world and conceptual entities (instances of the objects defined at the previous level) that constitute the actual models of specific production facilities.

2.1 Metamodel Primitives

The primitive objects that belong to the metamodel level are depicted in Figure 2 (See Figure 3 for notation conventions). They are: (1) Basic class, (2) Relationship and (3) Perspective. Basic class represents objects that allow the description of the application domain. A Basic class is characterized by a set of attributes and relationships that link it with other domain objects. At this level object behavior is not considered. Basic classes are, among others, Function, Task, Objective, Goal, Resource and Organizational Unit. Some considerations regarding the contents of Figure 2 have to be made since it expresses the cartesian product of Basic Class and Relationship. Without introducing any additional constraints this fact would allow, for instance, a Resource to be composed-of a Task. Therefore, some constraints have to be defined. For the sake of simplicity only two of them are shown at the right bottom of the model.
When modeling an organization, the developed models should reflect what happens in it. A production environment is far from being a static structure. For example, during product or process design activities, sequences of prototypes are generated till reaching final versions that meet the requirements imposed by different enterprise departments such as Production, Engineering, Finance, etc. Similarly, product specifications can be modified along product life cycle due to changes in customer requirements, quality improvements, recipe modifications, etc. In order to capture the previous ideas, an object version is defined as an alternative of a particular class. Two class objects are related by a “gives-rise-to” link to express the fact that both constitute different versions of the same entity.

A Relationship object links two class objects. It is identified by a label and has a minimum (0, 1) and maximum (1 or greater than 1) origin and destination cardinality. The name of a relationship describes the type of role that the destination object plays in relation to the origin one.

A role is defined as the ability or capability that an object has when it participates in a specific relationship with another object. In order to represent the different roles of an object the Perspective construction is used. As seen in Figure 2, Resource is defined as a composite object. Each Resource component has a perspective that represents information pertaining to a given role of the resource. The link that relates an object with its perspective is called “has-as-perspective”, and its inverse relationship is named “is-a-perspective-of”. A Resource may have zero or more perspectives, but a perspective is associated with only one Resource. Thus, the perspective idea allows resources to be seen in different ways according to the tasks they are involved into. For example, as seen in Figure 4(a), information concerning a product manufactured by a given company could be organized according to the following perspectives: (1) Product Market Data, that encapsulates information relative to price, sale conditions, type of package, etc.; (2) Product Inventory Data, that represents information regarding to storage conditions, actual level of stock, stock security level, etc.; (3) Product Planning Data, that maintains information on the allowed and average scrap levels, standard processing times, etc.; (4) Product Engineering Data, that summarizes data about product
formula, associated recipe, etc.; (5) Finished product; (6) Intermediate product and (7) Raw material. These last three perspectives take into account the production plant where the product is utilized/produced and the processes it is involved into. Figure 4(b) shows that raw materials, intermediate and finished products are organized in a composition hierarchy to represent the structure of a product. However, some constraints should be added to this metamodel to make it meaningful. For instance, an intermediate product is such if it is composed of either raw materials or other intermediate products.

**Figure 4**  Product Domain Models.

(a) Several views of the Product class   (b) Composition hierarchy representing a product structure.

The structure and behavior (if it were considered) of the objects identified in the different metamodels are given by a metaclass called Metaclass. According to Kiczales et al. (1991) different Metaclass specializations allow the definition of the basic object-oriented paradigm. Since Metaclass is not a concept that pertains to the domain being studied, it is not shown in the previous models. Thus, this paper focuses on the use of metamodels as templates employed to represent production environments, and it does not address the way metamodels are constructed as instances of higher level classes.

### 2.2 Enterprise Metamodel

One of the main metamodels presented in this paper, referred to as a "Corporate or Enterprise" metamodel, relates organizational units with functions (See Figure 5). In order to identify the system being studied an Organizational unit object is employed. This object represents the organization being analyzed in relation to other organizations, as well as its decomposition in different subsystems (division, department, plant, storage area, etc.). In this way, interactions with customers, raw material suppliers and service providers receive the same characterization as interactions between internal departments. Thus, this metamodel supports integration across the boundaries of the enterprise in the same way as it does internally. The proposed metamodel assumes that one or more responsibility lines are present. This is expressed through the "composed-of" relation shown in Figure 5; it links two Organizational unit objects and has a one or greater than one cardinality.

According to this metamodel, any Organization unit has two perspectives: client and server. Therefore, the whole enterprise can be abstractly seen as a service organization. For instance,
the Planning Department supplies services to (i) the Production Department, in the form of production plans, (ii) the Purchase Department, in the form of material requirements, etc. On the other hand, it acts as a client when it receives information about (i) sale forecasts and firm customer orders from the Sales Department, (ii) product recipes and bills of material from the Product Engineering Department and (iii) processing capacities and equipment availability from the Production Area. Services provided by an Organizational unit are carried out as Functions performed by it.

![Diagram of Enterprise metamodel](image)

**Figure 5** Enterprise metamodel.

*Functions* represent activities that are continuously carried out in a corporation such as Planning, Production, Research, Accounting, etc. The identification of Functions should represent fundamental concern for how the corporation operates. Though this metamodel links Functions with Organizational units by resorting to "is-performed-by" relationships, these links may change quite often since some corporations reorganize their structure frequently. Each Organizational unit performs one or more Functions, and each Function can be done by more than one Organizational unit. An object called Responsible-for-OU identifies the person that coordinates all the activities that are performed at the Organizational unit. A person can be responsible for more than one Organizational Unit; this is expressed by the one or more constraint at the origin of the responsible link.

The use of this metamodel assumes a transitivity relationship. Let A and B be two Organizational unit objects, such that A is composed-of B. If B performs Function X, then X is also performed by A. However, the converse relationship is not always true because an Organizational unit component not necessarily performs all the Functions associated with the whole unit.

### 2.3 Function-Task Metamodel

Another basic metamodel, the so-called Function-Task metamodel presented in Figure 6, focuses on functions. Identification of functions should be independent of the current organization chart. An organization may change but still have to carry out the same functions. Metamodel functions can be further decomposed ("composed-of" link). For example, the Engineering function may be decomposed into Product Development and Process Development.

The objectives of a Function are achieved through the sequential and/or parallel execution of Tasks. A Task can be defined as an operation or a specific thing to be done. Tasks differ from Functions from a time-related point of view. Whereas functions are continuously performed, tasks have precise starting and finishing times. At the Task level, all the resources
required and generated by a given task are explicitly represented. From this knowledge, all the resources associated to the accomplishment of a Function can be inferred.

![Figure 6](image)

**Figure 6** Function-Task metamodel.

A Task describes, in different abstraction levels, operations to be done. So, Tasks can be decomposed into subtasks, and these ones into new subtasks. The decomposition process is recursive; it finishes when the level of elementary tasks is reached (closure condition). The Task complexity degree determines the level of decomposition. In certain situations, there may be alternative ways in which a given Task can be solved. Therefore, a Task may have different decomposition structures. For example, a given scheduling task can be carried out differently if distinct goals are pursued. One decomposition structure is used if the reduction of scrap material goal is sought and another one if the reduction of work-in-process inventory is pursued.

According to the proposed metamodel, a Function is associated with a set of Objectives. An object called Objective identifies an aim or purpose of the organization. It can be pursued by more than one Function. The objectives associated with a Function are translated, at the level of tasks, into a set of Goals ("generates" relationship). Goals are specific targets that are intended to be reached by a given time. According to their time horizon they can be classified as strategic, tactic and operative.

Task decomposition structures provide an appropriate mechanism to represent the way in which tasks are solved. However, they are not sufficient to express coordination knowledge concerning the order in which subtasks are to be solved. Thus, establishing temporal links among tasks is very important. In this contribution, they are modeled by the temporal relationships proposed by Allen (1983). In order to represent them the temporal-relationship object was defined. It is specialized into the subclasses: before, equal, meets, overlaps, during, starts, and finishes. These links express precedence relationships among tasks as well as constraints that apply when resources are assigned to tasks.

Another main object in this metamodel is Resource that describes the elements that are created, consumed, used, produced, modified or eliminated by a Task during its execution in order to satisfy the task’s Goals. As seen in Figure 7 the metaclass Resource (belonging to the Metamodel layer) has been instantiated into the following classes: Information (data items, files, drawings, messages, etc.), Material-Utility-Supply (products, steam, parts, electricity,
etc.), Device (tools, machines, software packages, etc.) and General-resource (people, etc.), all of them belonging to the Model layer. Thus, in this layer, Resource makes reference to both information and physical resources. Resource instances can be further specialized. For example, Utility (Steam, Electricity, Compressed Air, Cooling Water, etc.) and Supply (Packaging Material, Lubricants, etc.) are subclasses of Material-Utility-Supply. Similarly, Device can be specialized into Equipment-item, Tool, Software Package, etc.

![Diagram](image)

**Figure 7** Some Resource instances in the Model Layer

The links that connect Tasks with Resources have been generalized by the task-resource-link object, that is a specialization of the Relationship Class. This object expresses the fact that a task may be related to one or more resources, and that a given resource is associated with one or more tasks. The degree of abstraction adopted to model a Task determines the degree of complexity in which the associated resources are represented. For example, at a given level of abstraction it may be considered that a task named Manufacture-product-A uses the Production-line-X resource. However, when this same task is modeled in greater detail the Production-line-X resource can be expressed in terms of its constituent equipment items.

As it was mentioned before, a link that connects a Resource with a Task defines a specific resource role in relation to the task. The meaning of the different task-resource classes that specialize task-resource-link (See Figure 8) is given by the following definitions:

- **uses.** It represents the fact that a resource is utilized during the time interval in which the task is performed. During the task execution period the resource changes its state, but the initial and final states are the same. This link is specialized into the employs relation which indicates that the resource is exclusively used by one task. For example, if an equipment item is employed by a task, it would suffer the following changes of state: idle - busy - idle.

- **contributes.** It is applicable to consumable resources, indicating that a given amount of a resource disappears when the task is finished. Since the resource consumption implies its transformation into new resources, this relationship is pertinent to any resource that satisfies conservation laws.

- **produces.** Indicates the creation of a new resource. It is the converse relation of consumes. So, it is also applicable to resource classes that satisfy conservation laws.

- **creates.** The use of this link implies the creation of a new resource without satisfying any conservation law. For example, this link may represent the generation of a new version of an existing information resource, such as a new product design, as it was previously discussed.

- **eliminates.** It is creates converse relationship. This relation indicates that the resource disappears definitively (without satisfying any conservation law) when the task is finished.

- **modifies.** This relationship indicates that, as a result of the task execution, the resource suffers a change of state.
The purpose of this section is to show the usage of the metamodel level primitives to represent the knowledge of specific production facilities. First, a series of generic models was constructed in the intermediate Model layer. These models can be made applicable to different types of organizations by properly specializing them.

In order to build these models specific questionnaires were generated. They were aimed at obtaining the knowledge necessary to describe the organization under analysis. Specifically, questionnaires try to identify: (i) Organizational units, (ii) Functions performed in the organization, (iii) Material and information flows linking the different functions, (iv) Information needs associated with the distinct hierarchical levels of the organization. In particular, for each function carried out in the production enterprise, questions such as the following were intended to be answered:

- main function objectives;
- lower level functions associated with it;
- tasks required to carry out each function;
- tasks' decomposition structures;
- tasks' precedence relationships;
- physical and information resources required by each task;
- physical and information resources generated by each task.

### 3.1 Example

Let us now consider the case of the XYZ production enterprise. Questionnaire answers revealed all the functions that are carried out by the six organizational units that comprise the XYZ company. One of functions was taken as an example because there is no available model for it in the Model layer. Then, it was necessary to instantiate a new one by using the Metamodel layer primitives. The chosen function is Production Planning and Control, that will be modeled as an instance of the Function-Task metamodel described in Figure 6.

A partial view of the resulting model is shown in Figure 9. The model provides a description of the function Production Planning and Control in terms of a hierarchy of functions: Production Control, Production Planning and Inventory Management. Production Planning consists of the subtasks (i) Create Aggregate Plan, (ii) Create Master Plan, (iii) Create
Material Requirements Plan (MRP), and (iv) Create Schedule. In addition, it expresses the fact that the generation of a schedule depends on the realization of the task Create MRP, and that the latter task has to be started after the completion of the Create Master Plan task ("before," link). The same idea applies to the two other tasks related by the "before" link. Figure 10 presents an explosion of the Create MRP task introduced in the previous model. According to this figure, after subsequent decompositions Create MRP originates the tasks Determine Gross Requirements, and Determine Net Requirements. Moreover, to completely describe Create MRP, Figure 10 shows the resources that are needed to carry out the task (Master Plan, Product Structure, Product Lead Times, etc.) as well as the outputs generated by it (Raw Material Purchasing Agenda, Final and Intermediate Products Production Agenda, etc.).

Having developed these generic models, the actual models of the XYZ company can be generated and values to all the object attributes can be assigned. It should be mentioned that, for the sake of clarity, models shown in Figures 9 and 10 do not include cardinality constraints.

Figure 9 A partial view of the Function-Task metamodel which describes the Production Planning and Control function.

4 CONCLUSIONS

This paper presents advances in the development of an open model architecture consisting of three layers: (i) a Metamodel layer, where metamodels are defined, (ii) a Model layer associated with production enterprise generic models and (iii) a Universe of Discourse layer related to real world and conceptual entities that constitute the actual models of specific production facilities. The work focuses on the introduction of task and domain metamodels (based on the Object Technology), that are being used as templates in the construction of a model library. The proposed architecture is said to be open because (i) Modeling primitives within the metamodel layer can be specialized, (ii) New generic models, pertaining to the intermediate level, can be generated via metamodel instantiation, (iii) New generic models can be created by specializing existing generic models. The proposed architecture is being tested by modeling a small food company.
Metamodels for information system modeling

Figure 10  Explosion of the high level function shown in Figure 9.

5 REFERENCES


6 BIOGRAPHIES

Gabriela S. Mannarino is a Research Fellow of “Consejo Nacional de Investigaciones Científicas y Técnicas” (CONICET) at INGAR (CONICET - Fundación ARCIEN). She holds an Information Systems Engineering Degree from “Universidad Tecnológica Nacional, Fac. Reg. Santa Fe” (1993). She is currently working on object oriented analysis and design methodologies and the development of integrated information systems for production environments.

Gabriela P. Henning is an Assistant Professor and Researcher at INTEC (CONICET-Universidad Nacional del Litoral), Santa Fe, Argentina. She received a Chemical Engineering Degree from “Universidad Tecnológica Nacional, Fac. Reg. Rosario” (Argentina) in 1981 and earned her Ph.D. in Chemical Engineering from “Universidad Nacional del Litoral” in 1986. She was a Postdoctoral Fellow at the Massachusetts Institute of Technology (1986-1989), specializing in Applications of Artificial Intelligence in Process Engineering. Her research interests include the application of knowledge-based and mathematical programming methodologies to industrial scheduling problems, the development of high-level languages for process engineering activities and the design of integrated information systems for production environments.

Horacio P. Leone holds positions as Associate Professor at the Information Systems Department, Universidad Tecnológica Nacional Fac. Reg. Santa Fe, and Researcher at INGAR (CONICET-Fundación ARCIEN), Santa Fe, Argentina. He received a Chemical Engineering Degree from “Universidad Tecnológica Nacional, Fac. Reg. Rosario” (Argentina) in 1981, and earned his Ph.D. in Chemical Engineering from “Universidad Nacional del Litoral” in 1986. He was a Postdoctoral Fellow at the Massachusetts Institute of Technology (1986-1989), specializing in Applications of Artificial Intelligence in Process Engineering. His current research focuses on object-oriented analysis and design of information systems for process industries and the application of artificial intelligence methodologies in decision support systems for fault diagnosis.
Abstract
Business re-engineering and enterprise integration efforts are supported very efficiently by enterprise modelling methodologies. However, with the number of methodologies available the comparison and selection of the most suited one becomes a rather difficult task. Most modelling methodologies orient themselves on the life-cycle concept but usually cover different part of the cycle itself. In addition, terminology and modelling constructs/language for representation of the model contents are further obstacles to be overcome in the selection process.

Representation of modelling methodologies as business processes enables their comparability in terms of enterprise life-cycle coverage and capability of enterprise information collection and representation. The paper presents the results of an analysis carried out for several enterprise modelling methodologies highlighting their similarities and differences. All modelling methodologies follow the enterprise life-cycle with emphasis on the requirements definition phase. Several methodologies carry enterprise modelling through design specification and implementation description to operation and model maintenance. Language expressiveness is quite different both in number of language constructs provided and their use in enterprise modelling.

In addition, the business process representation provides explicit identification of the information to be collected in the model. Both the information needed for the different modelling tasks and the results of the tasks can be explicitly identified thereby guiding the user of the methodology.

The analysis identifies the compatibilities of the different enterprise modelling methodologies and their emphasis on particular parts of the enterprise modelling task. It is hoped that this work also helps to harmonise the results of enterprise modelling as well as the terminology used. Both are very much needed in the work on enterprise integration.

Keywords
Enterprise Integration, Enterprise Architectures, Enterprise Modelling, Business Modelling, Modelling Methodologies, Modelling Languages.

1 INTRODUCTION
Methodology is the system of methods and principles used in a particular discipline. Method is a way of proceeding or doing something; the technique or arrangement of work for a particular field\(^1\).

\(^1\) Collins Dictionary and Thesaurus, 1987
These definitions imply the process nature of both methods and methodology. Process representations, especially graphical ones, are much more easily understood and comparable with each other. In addition, all of such methodologies are based on the life-cycle concept which allows a comparison of the different methods in terms of the coverage of different process steps in the life-cycle. The paper presents examples of the process representation of several Enterprise Modelling Methodologies. The graphical representation of the different methodologies as process models is based on CIMOSA an ESPRIT supported development.

The paper is intended to demonstrate the benefits of a common process oriented representation of modelling methodologies. It does not claim completeness and full correctness of the process models, which will need further work to capture all the details of the textual description available today.

The different methodologies represented and compared are ARIS², CIMOSA³, GRAI/GIM⁴, IEM⁵ and PERA⁶ with process models currently available only for CIMOSA, IEM and PERA. The work is based on material describing the different methodologies available to the author. It represents the authors view of the methodologies and may be modified in the course of further discussions with the developers and owners of the methodologies themselves. Due to the limitations of a paper only the example of the modelling methodologies with the widest life-cycle coverage (PERA) is presented with the graphical representation of its process model.

In addition, the paper compares the modelling languages used in the different methodologies. For more information on CIMOSA representation see references [1] and [2]. For a comparison of different methodologies see also references [3] and [4].

2 THE METHODOLOGIES - AN OVERVIEW

The different modelling methodologies have all been developed with different applications in mind. Therefore emphasis is on different aspects of enterprise modelling. Nevertheless they all contribute to enterprise integration and therefore should contribute to a common view on the subject. This paper tries to highlight the differences in goal and application areas of the different methodologies.

ARIS (ARchitecture for Information Systems) [5]
The ARIS focus is on the design of enterprise information systems. Therefore it provides specific modelling support for the Information Technology part of the enterprise (IT concept support). ARIS supports enterprise modelling from operation concept and IT concept to IT system implementation.

CIMOSA (CIM Open Systems Architecture) [1][2] CIMOSA models are intended to be used for operational support rather than as project guides in developing or re-engineering business entities. Operational use is understood as decision support for evaluating operational alternatives as well as model driven operation control and monitoring. CIMOSA supports the engineering of enterprise models from requirements definition to implementation description, their operational use and model maintenance supporting system changes and business re-engineering.

GRAI/GIM (Graphs with Results and Activities Interrelated/GRAI Integrated Methodology) [6]
GRAI was initially developed to model the decisional structure of a manufacturing enterprise for strategic, tactical and operational planning. GRAI was extended to support the design of CIM systems leading to GIM as an integrated methodology for business process modelling. With special emphasis on the decisional aspects, the concept (analysis), structure (user oriented design) and realisation (technical oriented design) phases of the life-cycle concept are supported.

---
² ARchitectur f"ur Informations Systeme (Architecture for Information Systems)
³ Open System Architecture for CIM
⁴ Graphe à Résultant et Activités Interreliées(Graphs with Results and Activities Interrelated)/GRAI Integrated Methodology
⁵ Integrated Enterprise Modelling
⁶ Purdue Enterprise Reference Architecture
Comparison of enterprise modelling methodologies

IEM (Integrated Enterprise Modelling) [4][10]
The IEM modelling methodology supports creation of enterprise models for business re-engineering and therefore allows also to model process dynamics for evaluation of operational alternatives. IEM supports the main phases of the enterprise life-cycle (requirements, design, implementation and model up-date).

PERA (Purdue Enterprise Reference Architecture) [7]
The PERA modelling methodology is intended to support and guide the development of the Master Plan for an Enterprise Business Entity. The methodology covers the complete project of introduction, implementation and operation of an enterprise business entity which may be either part of a larger entity or be the complete enterprise itself. The life-cycle starts with the definition of the Business Entity to be modelled, identifying its mission, vision, management philosophy, mandates, defines project sponsors, leaders and members, etc. and ends with obsolescence of the plant at the end of the operational phase.

3 PROCESS MODELS OF MODELLING METHODOLOGIES

The modelling methodologies are described in terms of their information exchange with the environment (CIMOSA Domains) and their internal process structure. The different processes (DP = Domain Process) identified correspond to the phases of the system life-cycle. These processes are further detailed as either sub-processes (BP = Business Process) or activities (EA = Enterprise Activity). Behavioural Rules define the process flow (control flow) identifying the conditions for continuation after ending an activity. Due to the space constraints of the paper the process model of only one of the methodologies (PERA) is presented (Figures 1 to 3). The information used and produced in the different activities is presented in Table 1.

This part of enterprise modelling allows to identify and provide/eliminate missing or redundant information and no value information, respectively. A comparison of the different methodologies (PERA, CIMOSA and IEM) is presented in Table 2 (at the end of the paper). The CIMOSA modelling methodology is described in a recent publication [8].

Process Model of PERA (Purdue Enterprise Reference Architecture)
The PERA modelling methodology covers the complete enterprise life-cycle starting from Business Entity Identification and ending with the turn-down of the plant at the end of the operational phase. Its life-cycle phases are described for personnel, information and product operational requirements leading to an information architecture, a human and organisational architecture and a manufacturing equipment architecture.

Process Representation of the PERA Modelling Methodology
The following is an attempt to establish a process model of the Purdue Enterprise Reference Architecture methodology using the CIMOSA modelling language (constructs). A draft of the process model is provided which has been developed in co-operation with T.J. Williams and co-workers. The modelling environment overview (Figure 1) provides the relation between the further detailed CIMOSA Domain ‘Enterprise Business Entity Master Plan Development’ and the none-CIMOSA Domains. Information exchange is identified on a rather high level indicating information and events exchanged between the CIMOSA Domain and the none-CIMOSA Domains.

PERA Process Model Overview
The details of the CIMOSA Domain are shown in Figure 2. Seven Domain Process have been defined covering each one of the different phases of the system life-cycle identified in the layering diagram of the PERA methodology. Enterprise Events have been defined which enable the cooperation of the different Domain Processes indicating completion of processes or needs for changes of results of previous ones. Figure 3 provides an example of the details of the different Domain Processes represented on Business Processes and Enterprise Activity level. The example shows the parallel efforts for the three architectures of PERA for information, human and organisation and manufacturing equipment. Behavioural Rules are only indicated but are not further defined.
PERA Information Identification

Representing the modelling methodology as a business process allows to identify the information used and produced by the different task. This can become the knowledge base of the enterprise ensuring a content which is identified as being both used and produced during enterprise operation.
Table 1 shows an example of the information needed and created by the PERA methodology. The different information objects described in the PERA literature have been structured into a set of enterprise objects (CIMOSA term) which present a part of a high level information model for the PERA methodology. The tasks which use and produce the information objects are indicated. Referring to the PERA literature this table indicates the consistency problems of textual descriptions.

Several of the information objects identified in the PERA methodology are either not used or not produced. Completing this table according to the business process representation at the necessary level of detail allows to identify all information and therefore provides a complete and consistent information model of the enterprise information used and produced during the modelling process. Providing real time maintenance for such an enterprise model will ensure an always up-to-date knowledge base of the enterprise.

<table>
<thead>
<tr>
<th>Management Mission, Vision and Values</th>
<th>Enterprise Object</th>
<th>Where Used</th>
<th>Where Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company Business</td>
<td>EA-20,21 (FII-2-1)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
<tr>
<td>Marketing Strategies</td>
<td>EA-22 (FII-2-1/II-3-4)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
<tr>
<td>Enterprise Business Entity Information</td>
<td>EA-1,23 (FII-1-1)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Goals and Objectives</th>
<th>Enterprise Object</th>
<th>Where Used</th>
<th>Where Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise Level</td>
<td>EA-1 (FII-3-4)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Internal Knowledge</th>
<th>Enterprise Object</th>
<th>Where Used</th>
<th>Where Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>EA-23,BP-3.2</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
<tr>
<td>Mfg. Capabilities</td>
<td>BP-3.2 (FII-2-1/2)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
<tr>
<td>Operational Policies</td>
<td>EA-23 (FII-2-2/8)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
<tr>
<td>Mfg Requirements</td>
<td>DP-4</td>
<td>FII-3-4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Internal Constraints</th>
<th>Enterprise Object</th>
<th>Where Used</th>
<th>Where Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical Success Factors</td>
<td>EA-22 (FII-2-1/2)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>External Knowledge</th>
<th>Enterprise Object</th>
<th>Where Used</th>
<th>Where Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>EA-23,BP-3.2 (FII-1-1)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
<tr>
<td>Experience of other Industries</td>
<td>DP-4 (FII-2-1)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
<tr>
<td>User Requirements</td>
<td>DP-4</td>
<td>FII-3-4</td>
<td></td>
</tr>
<tr>
<td>Customer Requirements</td>
<td>DP-4 (FII-2-2)</td>
<td>DM ‘Enterprise Management’</td>
<td></td>
</tr>
<tr>
<td>Legal Requirements</td>
<td>DP-3</td>
<td>FII-2-9</td>
<td></td>
</tr>
<tr>
<td>Standards</td>
<td>DP-3,4</td>
<td>FII-3-4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>External Constraints</th>
<th>Enterprise Object</th>
<th>Where Used</th>
<th>Where Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Economic Conditions</td>
<td>DP-2</td>
<td>FII-2-1/II-3-4</td>
<td></td>
</tr>
<tr>
<td>World Markets</td>
<td>DP-2</td>
<td>FII-3-4</td>
<td></td>
</tr>
<tr>
<td>Environment</td>
<td>DP-2</td>
<td>FII-3-4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Planning Data</th>
<th>Enterprise Object</th>
<th>Where Used</th>
<th>Where Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transition Plan</td>
<td>DP-3</td>
<td>FII-2-1/8</td>
<td></td>
</tr>
<tr>
<td>New Plan</td>
<td>DP-4</td>
<td>FII-2-1/8</td>
<td></td>
</tr>
</tbody>
</table>

Reference: DP/BP/EA (CIMOSA Process/Enterprise Activity) FII/II (PERA Figure)
Methodology Comparison PERA - CIMOSA - IEM
Table 2 shows the process models of the three methodologies at the Business Process level with identification of lower level Enterprise Activities. The latter is still to be done for the IEM modelling methodology. The number of activities identified for PERA and CIMOSA are 48 and 77 respectively demonstrating the higher level of details provided by CIMOSA. This is needed for the intended use of the CIMOSA model.

The representation follows the system life-cycle concept identified for the PERA methodology adding the maintenance phase of CIMOSA and IEM. This comparison demonstrates the advantage of the process oriented presentation of the modelling methodologies enabling direct comparison of the different methods in terms of coverage of the system life-cycle and different emphasis on the different phases.

4 MODELLING FRAMEWORK COMPARISON
A more global comparison of all modelling methodologies identified in this paper is shown in Tables 3.1 to 3.3. Using the Generalised Enterprise Reference Architecture and Methodologies (GERAM) [9] definition of the life-cycle phases the corresponding parts of the different methodologies have been identified. In addition to the life-cycle phases represented already in Table 2 for PERA, CIMOSA and IEM the Model Views and Genericity Levels are identified for the five methodologies investigated. The tables again indicate the terminology problem existing in enterprise modelling. But there is a surprisingly high level of terminology consistency.

Life-cycle Dimension
Table 3.1 indicates a rather similar coverage of the centre life-cycle phases (requirement, design, implementation) by all modelling methodologies. PERA covers the two uppermost GERAM layers

---

8 a ‘not defined’ entry means no formal identification exists. But the methodology may still provide specific solutions.
Comparison of enterprise modelling methodologies

for the identification of the Business Entity and definition of its management policies, etc. This information is assumed to be provided by enterprise management in all other methodologies.

The Operation Layer is explicitly defined in PERA only. Its existence in CIMOSA is recognised, but it is not seen as part of the modelling methodology. CIMOSA distinguishes between the enterprise engineering environment and the operation environment assuming models to be used as operational support (decision support tool) and directly in model driven operation control and monitoring. With this vision of enterprise model application, model maintenance is seen as a very important life-cycle phase, which is explicitly identified in both CIMOSA and IEM and contained in the operation layer of PERA.

The GRAI/GIM modelling framework distinguishes between the three architectural levels (Concept, Structure, Realisation) and three modelling activities (Analysis, User Oriented Design, Technical Oriented Design). The first two activities are relating to the first two architectural levels and the last activity is concerned with the realisation level. Two different sets of Model Views (see below) are identified for the different architectural levels.

Model View Dimension

Different views on the model help to reduce model complexity for the user. As shown in Table 3.2 such model views are provided by most methodologies, however, not all with the same capabilities. CIMOSA assumes one consistent enterprise model on which particular views are provided for the user in the engineering environment to allow for model engineering on a particular aspect of the enterprise operation (Function, Information Resource, Organisation, others tbd). ARIS provides a similar approach, but has identified the Control View for integrating the different views into a common process model. GRAI/GIM and PERA identify different views, but there is no real integration into one consistent model yet.

PERA changes its view concept across the life-cycle phases from a global view for the first and part of the second layer. It defines two views (Information Architecture and Manufacturing Architecture) for most of layer two and all of layer three. PERA continues thereafter with three views (Information Systems Architecture, Human & Organisation Architecture, Manufacturing Equipment Architecture).

GRAI/GIM identifies a unique Decision View which is at the centre of the GRAI methodology enabling modelling of strategic, tactical and operational planning.

IEM does not define model views explicitly but provides viewpoints on a common model. Therefore its modelling language constructs are related to the different views as well.

Genericity Level Dimension

This framework dimensions separates the particular model from the reference architecture which supports model creation. The reference architecture may contain generic building blocks or constructs for modelling (the words of the modelling language) and reference or partial models which may be used as macros in the modelling process. Except for PERA which only provides a single task module, all methodologies have a rather populated generic level and almost all provide sets of partial/reference models as well (Table 3.3).

5 MODELLING LANGUAGE CONSTRUCTS COMPARISON

A very extensive comparison between IEM and CIMOSA modelling constructs has been made jointly by the two originating teams in their efforts on trying to converge to a common modelling language. This comparison is described in a joined paper submitted to the European standardisation [10] which has lead to the ENV 12 204 the pre-standard on enterprise modelling constructs [11].

Tables 4.1 and 4.2 give an overview of the modelling languages provided by the different modelling methodologies. In addition to GERAM, which does not define any language constructs, the ENV 12 204 has been included as a reference. All methodologies provide some type of support for representation of the model contents. These languages consist of sets of generic constructs or building blocks to represent enterprise processes, activities, information, resources, organisation, etc. The constructs enable collection of relevant information allowing to describe the enterprise objects according to the modelling goal. Only PERA is not defining such modelling language but relies mainly on textual description of its methodology.
The modelling constructs can be associated to model views even if the may play a role in other views as well. In Table 4.1 and 4.2 the construct sets are structured according to their major role in enterprise modelling.

**General Definitions**
Most methodologies provide some structuring definitions in addition to the specific constructs. These definitions identify either the model contents (GRAI/GIM, PERA) or distinguish between model engineering and model use (CIMOSA).

**Function View related**
Constructs for function representation are provided by all methodologies with specialisations provided by CIMOSA and IEM. Both provide the process representation in the function view as well. ARIS has defined the control view for the representation of its process chains. Both GRAI/GIM and PERA do not offer modelling of the dynamic behaviour of its processes.

**Decision View related**
This view is only provided by GRAI/GIM. It allows to model the decision structure of the enterprise as well as to differentiating between different types of decisions (strategic, tactical, operational) by identifying different time horizon for the decisions. All other methodologies model decision making activities as parts of their (management oriented) business processes.

**Information View related**
ARIS, CIMOSA and IEM all provide a rich set of constructs for information modelling. Both ARIS and CIMOSA include IT oriented modelling constructs for modelling the IT system. ARIS provides additional IT oriented modelling constructs in the control view and in the organisation view. GRAI/GIM has defined two modelling constructs for information modelling using the Entity Relationship Approach for representation of the information model.

**Resource View related**
Constructs for the resource view exist in CIMOSA and IEM. ARIS is concerned mainly with IT resources which are described in the control, information and organisation view. The construct technical resources is used to describe all non-IT resources.

**Organisation View related**
The organisation view is populated in ARIS, CIMOSA and IEM. Whereas in ARIS resource organisational aspects are included in this view, CIMOSA uses the organisation view for identification of organisational aspects only. The main purpose in CIMOSA is to identify responsibilities and authorisation on all other enterprise objects (processes, information, resources) and to establish an escape mechanism for out of line situations. IEM uses a special class of its Resource Object for identifying organisation entities.

**Modelling Language Constructs Comparison ARIS - CIMOSA - GRAI/GIM - IEM- PERA**
Similar to the different aims of the different methods in terms of modelling results the expressiveness of the particular languages differ as well. Only CIMOSA has the vision of on an executable model for operation control and monitoring. Therefore its modelling language is a very expressive one. All other methodologies are focusing on particular situations from enterprise integration project descriptions (PERA), decision systems modelling and CIM system design (GRAI/GIM), information system design (ARIS) to business process re-engineering (IEM). Therefore their modelling languages are tuned to that particular application area resulting in more specialised constructs like ARIS (IT resource description), GRAI (decision view) and IEM (special object classes: Product, Order, Resource). On the other hand PERA is relying on textual description of its methodology providing only a construct for representation of task and its information inputs and outputs. Hopefully this comparison will result in more harmonisation of modelling languages both in their contents and their terminology.
6 SUMMARY

The analysis demonstrates the value of process oriented representation of modelling methodologies. It provides comparability far beyond textual description in terms of coverage of the modelling processes, the frameworks and the expressiveness of the modelling languages. Most importantly the process model allows to identify the information used and produced during model creation. This information will lead to a consistent knowledge base of the enterprise in the course of enterprise modelling.

More work is still required on the contents of the different methodologies to establish its consistent process models. Work which can only be done by or in co-operation with the authors of the methodologies. For the comparison of the modelling languages the different constructs have to be compared on the attribute level to allow for thorough evaluation. Work which has only be done with CIMOSA and IEM[6]. Also identification of the information used and produced in the course of model creation is still far from complete. This identification has the potential of much more consistent modelling of enterprise information. An aspect which will increase the operational use of enterprise models considerably e.g. for decision support. If the knowledge base is kept consistent and up-to-date planning activities, evaluation of alternatives and investment decision will be based on current rather than historic information.

Additional benefits will be obtained by taking advantage of the common representation and converging terminology and task definitions. Today there is no common understanding on enterprise models and relating models from different enterprises is a rather difficult if not impossible task.

Even with the reasons accepted for the different methodologies, the need of compatibility remains for the user of enterprise modelling methodologies. Otherwise enterprise co-operation across organisation boundaries will not move into a really integrated mode and inter enterprise integration will never become a reality. A reality which is very much desirable for joint ventures and subcontractors or for their more modern versions of extended and virtual agile enterprises.

7 REFERENCES

## Table 2 Methodology Comparison - PERA - CIMOSA - IEM

<table>
<thead>
<tr>
<th>PERA</th>
<th>CIMOSA</th>
<th>IEM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DP-1 EBE Identification</strong></td>
<td><strong>BP-1</strong></td>
<td><strong>BP-1.1 - BP-1.7</strong></td>
</tr>
<tr>
<td><strong>EA-1</strong> identify Enterprise Business Entity</td>
<td>EA-73 - EA-143</td>
<td>(BP-1.1 - BP-1.8)</td>
</tr>
<tr>
<td><strong>BP-2.1 Management/Business Policy</strong>&lt;br&gt;Identification; <strong>EA-21 - EA-23</strong></td>
<td><strong>BP-2</strong></td>
<td><strong>EA-1 - EA-64</strong></td>
</tr>
<tr>
<td><strong>BP-2.2 Production Policy Definition</strong>&lt;br&gt;Definition; <strong>EA-24 - EA-26</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-3.1 Management/Business</strong>&lt;br&gt;Requirements Definition; <strong>EA-31-EA-33</strong></td>
<td><strong>BP-3.2</strong></td>
<td><strong>BP-2.1 - BP-2.3</strong></td>
</tr>
<tr>
<td><strong>BP-3.2 Production Requirements</strong>&lt;br&gt;Definition; <strong>EA-34 - EA-36</strong></td>
<td><strong>BP-1.1 - BP-1.7</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-3.2 Production Policy Definition</strong>&lt;br&gt;Definition; <strong>EA-21 - EA-26</strong></td>
<td><strong>BP-1.1 - BP-1.7</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-3.3 Manufacturing Equipment Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-41, EA-42, EA-47</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-4.1 Information System Architecture Design</strong>&lt;br&gt;Definition; <strong>EA-44 - EA-46</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-4.2 Human &amp; Organisation Architecture Design</strong>&lt;br&gt;Definition; <strong>EA-41 - EA-43</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-4.3 Manufacturing Equipment Architecture Design</strong>&lt;br&gt;Definition; <strong>EA-41 - EA-43</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-5.1 Information System Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-54 - EA-56</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-5.2 Human &amp; Organisation Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-51 - EA-53</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-5.3 Manufacturing Equipment Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-57 - EA-59</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-6.1 Information System Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-61 - EA-64</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-6.2 Human &amp; Organisation Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-65 - EA-66</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-6.3 Manufacturing Equipment Architecture Operation</strong>&lt;br&gt;Definition; <strong>EA-67 - EA-69</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-7.1 Information System Architecture Operation</strong>&lt;br&gt;Definition; <strong>EA-71 - EA-73</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-7.2 Human &amp; Organisation Architecture Operation</strong>&lt;br&gt;Definition; <strong>EA-74 - EA-76</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-7.3 Manufacturing Equipment Architecture Operation</strong>&lt;br&gt;Definition; <strong>EA-77 - EA-79</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-8.1 Information System Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-81 - EA-83</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-8.2 Human &amp; Organisation Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-84 - EA-86</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-8.3 Manufacturing Equipment Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-87 - EA-89</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-9.1 Information System Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-91 - EA-93</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-9.2 Human &amp; Organisation Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-94 - EA-96</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-9.3 Manufacturing Equipment Architecture Operation</strong>&lt;br&gt;Definition; <strong>EA-97 - EA-99</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-10.1 Information System Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-101 - EA-103</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-10.2 Human &amp; Organisation Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-104 - EA-106</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-10.3 Manufacturing Equipment Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-107 - EA-109</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-11.1 Information System Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-111 - EA-113</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-11.2 Human &amp; Organisation Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-114 - EA-116</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-11.3 Manufacturing Equipment Architecture Operation</strong>&lt;br&gt;Definition; <strong>EA-117 - EA-119</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-12.1 Information System Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-121 - EA-123</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-12.2 Human &amp; Organisation Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-124 - EA-126</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-12.3 Manufacturing Equipment Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-127 - EA-129</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-13.1 Information System Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-131 - EA-133</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-13.2 Human &amp; Organisation Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-134 - EA-136</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-13.3 Manufacturing Equipment Architecture Operation</strong>&lt;br&gt;Definition; <strong>EA-137 - EA-139</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-14.1 Information System Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-141 - EA-143</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-14.2 Human &amp; Organisation Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-144 - EA-146</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-14.3 Manufacturing Equipment Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-147 - EA-149</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-15.1 Information System Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-151 - EA-153</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-15.2 Human &amp; Organisation Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-154 - EA-156</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-15.3 Manufacturing Equipment Architecture Operation</strong>&lt;br&gt;Definition; <strong>EA-157 - EA-159</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-16.1 Information System Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-161 - EA-163</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-16.2 Human &amp; Organisation Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-164 - EA-166</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-16.3 Manufacturing Equipment Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-167 - EA-169</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-17.1 Information System Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-171 - EA-173</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-17.2 Human &amp; Organisation Architecture Manifestation</strong>&lt;br&gt;Definition; <strong>EA-174 - EA-176</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-17.3 Manufacturing Equipment Architecture Operation</strong>&lt;br&gt;Definition; <strong>EA-177 - EA-179</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-18.1 Information System Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-181 - EA-183</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-18.2 Human &amp; Organisation Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-184 - EA-186</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BP-18.3 Manufacturing Equipment Architecture Detailed Design</strong>&lt;br&gt;Definition; <strong>EA-187 - EA-189</strong></td>
<td><strong>BP-2.1 - BP-2.8</strong></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3.1 Modelling Framework Comparison - Life-cycle (Modelling Levels) - GERAM - ARIS - CIMOSA - GRAI/GIM - IEM - PERA

<table>
<thead>
<tr>
<th>GERAM</th>
<th>ARIS</th>
<th>CIMOSA</th>
<th>GRAI/GIM</th>
<th>IEM</th>
<th>PERA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
<td>EBE Identification (Enterprise Business Entity)</td>
</tr>
<tr>
<td>Concept</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
<td>EBE Concept Layer</td>
</tr>
<tr>
<td>Requirement</td>
<td>Operation Concept</td>
<td>Requirements Definition</td>
<td>Concept Level Analysis</td>
<td>Requirements Definition</td>
<td>EBE Definition Layer</td>
</tr>
<tr>
<td>Design</td>
<td>IT System Concept</td>
<td>Design Specification</td>
<td>Structure Level</td>
<td>System Design</td>
<td>EBE Specification Layer</td>
</tr>
<tr>
<td>Implementation</td>
<td>Implementation</td>
<td>Implementation Description</td>
<td>Realisation Level</td>
<td>Implementation Description</td>
<td>EBE Manifestation Layer</td>
</tr>
<tr>
<td>Operation</td>
<td>(Operation)</td>
<td></td>
<td></td>
<td></td>
<td>EBE Operation Layer</td>
</tr>
<tr>
<td>System Change</td>
<td>Model Maintenance</td>
<td></td>
<td></td>
<td></td>
<td>Model Up-Date</td>
</tr>
</tbody>
</table>

### Table 3.2 Modelling Framework Comparison - Model Views - GERAM - ARIS - CIMOSA - GRAI/GIM - IEM - PERA

<table>
<thead>
<tr>
<th>GERAM</th>
<th>ARIS</th>
<th>CIMOSA</th>
<th>GRAI/GIM</th>
<th>IEM</th>
<th>PERA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Function View (static)</td>
<td>Function View (static)</td>
<td>Function View (static)</td>
<td>Function Model View</td>
<td>Information Architecture. Information Syst. Arch.. (dynamic not defined)</td>
</tr>
<tr>
<td></td>
<td>Control View (dynamic)</td>
<td>Function View (dynamic)</td>
<td>Information Techn. View</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Information</td>
<td>Data View</td>
<td>Information View</td>
<td>Information View</td>
<td>Information Model View</td>
<td>not defined</td>
</tr>
<tr>
<td>Decision/Organisation</td>
<td>not defined</td>
<td>not defined</td>
<td>Decision View.</td>
<td>not defined</td>
<td>Human and Organis. Arch.</td>
</tr>
<tr>
<td></td>
<td>Organisation View</td>
<td>Organisation View</td>
<td>Physical View.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Manufact. Techn. View</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3.3 Modelling Framework Comparison - Genericity Levels - GERAM - ARIS - CIMOSA - GRAI/GIM - IEM - PERA

<table>
<thead>
<tr>
<th>GERAM</th>
<th>ARIS</th>
<th>CIMOSA</th>
<th>GRAI/GIM</th>
<th>IEM</th>
<th>PERA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generic</td>
<td>Generic</td>
<td>Generic</td>
<td>Generic</td>
<td>Generic</td>
<td>not defined</td>
</tr>
<tr>
<td>Partial</td>
<td>Reference Models</td>
<td>Partial</td>
<td>4 Levels of Abstract</td>
<td>Reference</td>
<td>not defined</td>
</tr>
<tr>
<td>Particular</td>
<td>Particular</td>
<td>Particular</td>
<td>Particular</td>
<td>Partial</td>
<td>not defined</td>
</tr>
</tbody>
</table>

### Table 4.1 Modelling Language/Construct Comparison - Function, Control and Decision Views - ENV 12 204 ARIS - CIMOSA - GRAI/GIM - IEM - PERA

<table>
<thead>
<tr>
<th>GERAM</th>
<th>ENV 12204</th>
<th>ARIS</th>
<th>CIMOSA</th>
<th>GRAI/GIM</th>
<th>IEM</th>
<th>PERA</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Definitions</td>
<td>not defined</td>
<td>not defined</td>
<td>Engineering Environment, Operation Environment</td>
<td>Decisional System, Information System, Physical System, Business Domain</td>
<td>not defined</td>
<td>Enterprise Business Entity</td>
</tr>
<tr>
<td>Modelling Constructs - Function View related (static)</td>
<td>not defined</td>
<td>Enterprise Activity</td>
<td>Function</td>
<td>Domain, Enterprise Activity, (Funct. Operation)</td>
<td>IDEF0 Activity</td>
<td>Activity, Function, (Action)</td>
</tr>
<tr>
<td>Modelling Constructs - Function View related (dynamic)</td>
<td>not defined</td>
<td>Business Process, Event, (Sequential Relationships)</td>
<td>Process Chain, Event, (Connectors) Cluster</td>
<td>Process (DP/BP), Event, (Behavioural Rules)</td>
<td>not defined</td>
<td>Function Chain, Funct Auton Unit, (Connecting Constructs)</td>
</tr>
<tr>
<td>Modelling Constructs - Decision/Organisation View related</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
<td>not defined</td>
</tr>
<tr>
<td>number of constructs</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>
Table 4.2: Modelling Language/Construct Comparison Information, Resource and Organisation Views - ENV 12 204 - ARIS - CIMOSA - GRAI/GIM - IEM - PERA

<table>
<thead>
<tr>
<th>GERAM</th>
<th>ENV 12204</th>
<th>ARIS</th>
<th>CIMOSA</th>
<th>GRAI/GIM</th>
<th>IEM</th>
<th>PERA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modelling Constructs - (Decision)/Organisation View related</td>
<td>Organisation Unit</td>
<td>Organisation Level, Organisation Unit, Attribute, Location, Network, Network Node, Network Unit, Technical Resource</td>
<td>Organisation Cell, Organisation Unit, (Organisation Element)</td>
<td>not defined</td>
<td>Object Class: Special Resource</td>
<td>not defined</td>
</tr>
<tr>
<td>Modelling Constructs - Information View related</td>
<td>Enterprise Object, Product, Order, Object View, Relation</td>
<td>Entity, Attribute, Relation, Terminology, Table, (Cardinality (ext.)), (Operators:) Classification Generalisation Specialisation Aggregation Grouping</td>
<td>Enterprise Object, Object View, (Inform. Element), Relation, (Cardinality), (Operators:) Classification Generalisation Specialisation Aggregation</td>
<td>Inform. Model, Entity, Relation</td>
<td>Object Class: Order, Product, Relation, (Operators:) Classification Generalisation Specialisation Aggregation</td>
<td>not defined</td>
</tr>
</tbody>
</table>

| From table 4.1 | 3 | 4 | 4 | 6 | 5 | 1 |
| Total number of constr. | 11 | 17 | 11 | 9 | 10 | 1 |
Bottom-up Modelling with CIMOSA

W. Reithofer  
Institute for Real-Time Computer Systems and Robotics,  
University of Karlsruhe, D-76128 Karlsruhe, Germany  
e-mail: rhofer@ira.uka.de

Abstract

CIMOSA enterprise models provide a basis for documenting, operating, and redesigning a company. However, up to now CIMOSA is not commonly accepted in industries. This is mainly due to the fact that CIMOSA users do not know how to create their particular models efficiently. Currently, they are faced with the collection of a tremendous amount of data which makes the model creation process an unaffordable effort. This paper proposes a new bottom-up modelling methodology for the design of particular CIMOSA models: The modelling starts with the identification of the most important process according to a specific goal. This core model is then expanded from inside to outside by adding further processes which are also relevant. With respect to the implementation of a corresponding modelling tool, the basic data structures for the model representation and the procedures which are needed to process them are also shown. Finally, our modelling and simulation prototype tool, the Virtual Factory Lab Victor is introduced.

Keywords

CIMOSA, Enterprise Modelling, Simulation

1 INTRODUCTION

Two of the main problems of today's enterprises are:

- **Enterprise redesign**: To replace the classical, function-oriented processes by product-oriented, computer integrated processes, enterprises are redesigned. The starting point of such a project is usually a thorough investigation of the "as-is" situation. The "as-is" processes are analyzed and redesigned to remove inconsistencies and to make them as straightforward as possible. Afterwards, the resulting "to-be" processes must be implemented in the physical system (see, e.g., [McMenamin, 1988]).
• **(Re-)Design of virtual enterprises**: A Virtual Enterprise [Rembold, 1994], [Goldman 1995] or Extended Enterprise [Browne, 1994] is a temporary alliance of several distributed work units. These units cooperate to manufacture particular products to meet new market needs very rapidly. The work units may be units of one or several enterprises. One individual enterprise may be involved in several virtual enterprises with different partners at the same time. The most important benefit of a virtual enterprise is the accumulation of core competencies of the involved partners and the reduction of time-to-market by optimally using existing production resources. Virtual enterprises already exist, but their formation, operation, reformation or dissolution is currently very costly in terms of management and administration. The (re-)design of a virtual enterprise is mainly concerned with the problem of assembling a consortium, splitting up the common task into subtasks and assigning the sub-tasks to the partners.

A model which could help to solve these problems, must represent the most important aspects of an enterprise like objects, functions and processes. Furthermore, particular models of individual enterprises should have the same structure to be comparable and reusable. This can be achieved by deriving individual models from a generic meta model.

Several existing enterprise meta models or reference architectures fulfill these requirements. A comparison of them can be found in [CEN, 1994]. Common to most enterprise models is that they integrate basic models for describing the various aspects, e.g., SADT for describing functions, entity attribute representations for describing objects etc. We work with the Open Systems Architecture for Computer Integrated Manufacturing Systems CIMOSA [AMICE 1994], [Formal Reference Base, 1994]. CIMOSA is based on a formal, properly defined model representation. Most parts of CIMOSA have become a standard (ENV 40.003, Framework for Enterprise Modelling). However, one has to admit that until now CIMOSA has not been generally accepted in industries. When designing an individual model, companies are faced with an immense effort which they cannot fulfill within an acceptable time span. Of course, they do not want to spend a lot of resources to develop a model which is already out dated before it is finished. The reasons for this huge modelling effort are the weak description of the modelling process, and the lack of tools which support the model creation and model validation.

## 2 REVIEW OF EXISTING WORK

### 2.1 The CIMOSA business modelling process

Apart from the model itself, the [Formal Reference Base, 1994] describes the CIMOSA model creation process. This description explains how to create a particular model from the reference architecture. The modelling process is decomposed into the three subprocesses Instantiation, Derivation and Generation. The Instantiation process expresses the relations between the three levels of genericness. The Generation process expresses the relations between the four CIMOSA views. The CIMOSA views may be seen as a filter which focuses, e.g., on the functional aspects or the information aspects of the CIMOSA enterprise model. The relations of the three modelling levels are described by the Derivation process. For the CIMOSA users who want to create their particular models, the instantiation process is the most important one. The [Formal Reference Base, 1994] gives a number of rules and constraints which must be taken
into account when creating particular models. However, these guidelines are completely unsuitable to help CIMOSA users to create models of their own, already existing systems.

A more concrete methodology is given in the *User Guide on Requirements Definition Model* [AMICE, 1994]. This modelling guide suggests a global top-down modelling process which starts with the identification of the domains, the definition of domain objectives and constraints, and the identifications of relationships between the domains. Afterwards, the domains are decomposed into domain processes. Domain processes are further decomposed into business processes / enterprise activities. The modelling process ends with the operational and information analysis. This course of actions has been demonstrated with an example in [Zelm, 1995]. After applying this procedure to practical examples, we found out that it is inefficient to model existing systems because of the following reasons:

- The top-down analysis cannot be strictly goal-oriented. It is burdened with the collection of irrelevant data since one cannot decide at an upper level, which, e.g., domain/business processes will be relevant for the solution of the given problem.
- The top-down analysis implies a lot of backtracking. Decisions made at an upper level can not be changed at a lower level.
- Top-down approaches are useful for system design from scratch. However, they are not recommended for the analysis of existing systems since the boundaries of the basic building blocks, i.e., the enterprise activities are already fixed.

The existing tools which support the design of particular CIMOSA models like, e.g., GTVOICE [VOICE, 1994] or SEW-OSA [Aguiar, 1995] implement this top-down modelling approach.

In summary, a modelling methodology is missing which allows CIMOSA users to create particular models in an efficient and goal-oriented manner, focusing on the processes and activities which are relevant to solve a given problem.

### 2.2 CIMOSA Modelling and Simulation Tools

Before using a particular model for problem solving, it has to be proven that the relevant aspects of the system are represented correctly. Simulation is suitable to support this task. The Formal Reference Base refers numerously to simulation. However, it is not described how to translate a particular CIMOSA model into a simulation model. Looking at existing CIMOSA modelling tools with simulation capabilities like, e.g., SEW-OSA [Aguiar, 1995], ARTIFEX [Bruno, 1994] and McCIM [Didic, 1993], we found out that they focus on specific applications. A problem which has not been solved by these tools is the on-line assignment, the so-called late binding, of resources to functions which are competing for the resources. Another problem is the simulation of processes where data is produced by a function and used by a subsequent function as control input.

### 3 A BOTTOM-UP MODELLING APPROACH

Our bottom-up modelling process starts with the design of a model of the most important process and expands this model by processes of decreasing importance. Since the modelling should be supported by a tool, we also have to take care about the data structures which repre-
sent the model and the procedures which are needed to combine the basic model elements bottom-up to more complex model components. According to the derivation process [AMICE, 1994], the model of a CIM system is created at three different levels of definition and expression: Each modelling level contains an internal structure consisting of three levels of genericness and can be accessed via four views. The modelling levels are the Requirements Definition Modelling (RDM) level, the Design Specification Modelling (DSM) level and the Implementation Description Modelling (IDM) level.

The problems described in the introduction of this paper can be mainly solved at the RDM and DSM level. The RDM level is not sufficient since information about operation times and resource capacities is needed to perform an analysis of efficiency of the various design alternatives. This information is given on the DSM level. The result of our modelling task is the selected design specification model.

3.1 The bottom-up modelling process

Our proposed bottom-up modelling process can be described roughly with the following procedure:

1. Goal definition
2. Definition of system boundaries and initial object
   3. Identification of functions applied to the initial object from its "birth" to its "death"
   4. Definition of the In/Outputs of the functions
   5. Identification of further relevant objects
   6. Completion of processes applied to further identified objects according to 3.-5.
3. Synchronize the processes and determine the triggering conditions

Each step of this procedure will now be described with an example. The example is taken from an industrial test case: A machine tool producing company has severe problems with the delivery of their products because of delays in the parts production. To find out the reasons of the delays, we performed a simulation study of the material flow [Roessel, 1995]. Apart from this conventional approach, we designed a CIMOSA model of the "as-is" situation.

The first step of our modelling process is the definition of the goal according to the problem we want to solve. In our example, the goal is to increase the due-date dependability of the parts production. In step 2, the boundaries of the system and the initial object are defined. In our case, the initial object is a single part or a lot of single parts. The system we want to improve is the mechanical work shop for parts manufacturing. The mechanical work shop is a job shop which consists of about 20 machine tools. The production of a part starts when its raw material and its work plan is available in the input buffer of the mechanical workshop. From here the object and the work plan are transported to the input buffer of the machine where the first operation will be performed. After the execution of this operation, the machine operator places the object and the work plan in the output buffer of the machine. Next, the object and the work plan are transported again to the subsequent machine and so forth until all operations of the work plan have been executed. Afterwards, the part is buffered in the output store of the mechanical workshop. Therefore, following step 3 of our modelling process, all functions which are applied to the object between its birth and death are transport and manufacturing operations. In general, the birth of an object may be either its entering of the system (in our example the raw
material) or its first appearance, i.e., the creation of the object. On the other hand, the death of an object may be its leaving from the system (example: finished parts stored in the output buffer from where they are handed over to the assembly department) or its disappearance because of, e.g., dismantling. The ordering of the functions according to their order of execution results in the initial process (Figure 1). The function transport has two ending statuses. Only if the goal of the transport operation is the output buffer of the mechanical workshop, the process terminates.

![Figure 1 Part Process.](image)

In step 4 the in/outputs of each function are defined (Figure 1). Both functions of our example manipulate the object part. Therefore, we defined an enterprise object part (EO-1) and an object view (OV-1) on this enterprise object which is used as input and as output of the functions. Additionally, both functions need a work plan. Consequently, an enterprise object "work plan" (EO-2) and corresponding object views (OV-2, OV-3) are defined. In the following this core model is expanded from the inside to the outside by adding further relevant processes. When we specified the functions, we found out that a suitable tool must be also available before a manufacturing operation can be started. Since the lack of available tools may influence the due-date dependability of the parts production, we decided in step 5 that tools are relevant objects (EO-3). We also created a model of the tool process (Figure 2). A tool process starts with the birth, i.e., the assembly of a complex tool out of its basic parts and ends with the death, i.e., the dismantling of the tool. After the assembly, the tool is transported to the machine where the manufacturing operation is performed. This is done by the operator of the machine. After executing the manufacturing operation, he brings back the tool to the central tool assembly work place. The dismantling of the tool is equal to its death. For synchronizing the tools and the parts process events are needed (EV-1, EV-2, EV-3). Event EV-1 is sent out after the tool has been transported to the machine. In conjunction with the ending status nextOp of the transport operation, this event triggers the execution of a manufacturing operation. EV-2 and EV-3 are used to trigger the "use" and the "transport back to work place" functions of the tool process.

In our example, we did not add any further processes. Other objects like, e.g., NC-programs are available when they are needed. Therefore, they are not considered to be relevant objects.

The final step in the design of the "as-is" model is the definition of triggering conditions of the processes (Figure 3). The part process starts, when the raw material and the work plan is available. After finishing the part process, a signal is sent to the domain "assembly". The as-
Bottom-up modelling with CIMOSA

Assembly of tools is triggered by the planning department. They send the assembly plans of the tools to the mechanical workshop independently of the work plans.

Figure 2 Tool Process.

We applied the proposed modelling methodology also to another example at the Rodenstock group [Limberger, 1996]. Here, by comparing the existing with the expected inputs of the modelled functions we discovered an error in the raw material order process. Its elimination led to a significant improvement of the raw material supply.

Figure 3 Triggering Conditions of the Processes.

3.2 Use of Particular Model Components (PMCs)

This section describes how we work with Particular Model Components (PMCs). The combination of PMCs is fundamental for the implementation of a tool which supports the bottom-up design of particular models. We defined five PMC types:
- **Enterprise Activity Type:** PMCs of this type represent the functions of an enterprise. They are the lowest level building blocks.
- **Business Process Type:** Is a sequence of Enterprise Activities / Business Processes.
- **Domain Process Type:** Represents a process which consists of Enterprise Activities and/or Business Processes.
- **Domain Type:** Represents the considered system.
- **Resource Type:** Occurrences of this PMC types actually perform the functions.

![PMC Type Enterprise Activity Diagram](image)

**Figure 4 PMC Type Enterprise Activity.**

Each PMC contains attributes and references. The attributes do not cause any problems. However, every reference is a pointer to another CIMOSA object which itself contains references to other objects as shown in Fig. 4 for a PMC of type enterprise activity.

The graph (Figure 4) contains three types of elements:
- **CIMOSA objects** (displayed with shaded polygons),
- **subgraphs** representing other PMCs (displayed with ovals), and
- **references** (displayed with arrows).

The CIMOSA objects are characterized by three rows. The first row gives the name of the object, the second all attributes of the object and the third all references to other objects. Every item of the second and third row is displayed within a rectangle. Two small rectangles are located on the right side above the item. The upper one indicates if the item exists (rectangle filled black) or does not exist (rectangle filled white) on the RDM level, whereas the lower one gives the same information for the DSM level. The items of the third row also contain numbers which give the cardinality of the references.
A PMC contains two types of references: *Internal references* point to an object which is a part of the considered PMC, whereas *external references* point to objects outside.

After the definition of PMC types, a set of rules is needed to define the combination of PMCs. For example, if a number of enterprise activities have been identified in the third step of our bottom-up modelling approach, they have to be ordered to form a process. This is done by instantiating a process template and updating it according to the modelled process. To allow the reuse of PMCs from one or several existing enterprise models for the, e.g., design of a model of a virtual enterprise it is necessary to implement the *export* and *import* functions for PMCs. These functions can be compared with identically named functions in a CAD system, but they are much more difficult to implement since they work on very complex structures, as shown in Figure 4. When joining PMCs, the following issues have to be taken into account:

- **Unambiguous object identification**: Each object of the model requires a unique identifier. Therefore, it may be necessary to change the identifiers of some objects.
- **PMC completeness**: To avoid the lack of referenced objects after PMCs have been combined, each PMC should be complete, i.e., it must not contain external references. External references can be avoided by either removing the reference (e.g. in case of an enterprise activity type component, the „Where used“ reference is removed) or including the referenced object in the PMC.
- **Consistency**: The model representation in CIMOSA is ambiguous, i.e., objects can be modelled in different ways. This fact causes problems if one wants to detect objects which should have the same semantic. For example, inserting an enterprise activity into a business process may result in references pointing to different object views which should actually describe the same object. To ensure the semantic consistency of the designed model, these doubled objects have to be compared and matched or in the case that they differ considerably from each other, the join operation must be refused. Semantic inconsistency occurs if, e.g., two enterprise activities should be connected, whereby the output of the first one is not the expected input of the following one.

### 3.3 Simulation of particular CIMOSA models

Simulation is a suitable way to validate the behavioral aspects of CIMOSA models. For example the model of the parts manufacturing system which has been created in section 3.1 can be validated as follows: After translating it into a simulation model it is fed in with recorded data, e.g., real work plans, tool plans, etc. of the machine tool company. Afterwards, simulation experiments are executed. The result of these experiments, i.e. the simulated start and end dates of the manufacturing operations are compared with the real system's behavior. Based on this comparison, one can decide whether the model is valid or has to be improved.

Our approach for simulating CIMOSA models is depicted in Figure 5. A pattern for each process instance in the CIMOSA model is derived. In our example we have two process patterns: The parts process and the tool process. Furthermore, every resource is represented by a corresponding instance. During run-time of the simulation, an occurrence of each process instance is created for every order. Each active process tries to progress. However, a process is delayed if the resources needed are not available. All active processes compete for the resources.
4 VIRTUAL FACTORY LAB KARLSRUHE

Victor consists of a modelling environment for the design of CIMOSA particular models and a simulation system for model execution (Figure 6).

It is implemented as an application specific building block library for the simulation system SIMPLE++[AESOP, 1994]. The building block library consist of building blocks for the CIMOSA objects Domain, Domain Relationship, Domain Process, Business Process, Enterprise Activity, Event, Declarative Rule, Integrity Rule, Enterprise Object, Object View, and Capability Set. We have also implemented a enterprise modelling frame (Figure 7).
For designing a particular CIMOSA model, an instance of this empty enterprise model is created. Afterwards, instances of building blocks representing the several CIMOSA objects are inserted directly into the subnets depicted with the $\Sigma$s. The modelling can start anywhere. It is possible to start with the definition of domains as well as with the modelling of enterprise activities or enterprise objects. The user is not forced to follow any modelling methodology. The referential consistency of the model is guaranteed by the tool. When defining a link between to objects (e.g. "comprises", "where used"), all references are updated automatically. To make the tool as flexible as possible, we implemented methods to convert an enterprise activity into a business process and a business process into a domain process and vice versa. It is also possible to aggregate and decompose elements of a process. The model can also be accessed via the four CIMOSA views (see Figure 7, buttons at the lower left corner). The function view has two modes: specification and decomposition. In specification mode, the opening of a building block allows to specify the properties of the object, i.e., to edit its template whereas in decomposition mode all objects and processes included are shown.

Particularly for the design of models of virtual enterprises and for reusing PMCs we implemented export and import functions.

![CIMOSA Modelling environment with Simple++](image)

**Figure 7** CIMOSA Modelling environment with Simple++.

Finally, Victor offers the possibility to transform the designed enterprise models into executable simulation models.
5 CONCLUSIONS

Although CIMOSA is one of the most powerful approaches in enterprise modelling, it is not generally accepted in industries. For them, the creation of a particular CIMOSA model is burdened with the collection of an immense amount of data. This makes the model creation process an unaffordable effort. The reason is that the CIMOSA business modelling process is not suitable to create a model in a goal-driven, efficient manner. Therefore, a new modelling methodology is proposed. The bottom-up modelling process starts with the definition of the goal. According to this goal, the system and the initial object are defined. In a third step, all functions applied to the initial object are identified and arranged in the core process. Other processes for further relevant objects are added to the core process. The advantage of this procedure is, that only relevant aspects of the system are modelled and that the model is created from inside (the core process) to outside (less important processes). To support this modelling process with a tool, rules for particular model component management are needed. These rules define how particular model components of five basic types can be linked together. Furthermore, a brief overview about the simulation of the created model is given. Finally, the Virtual Factory Lab Karlsruhe (Victor) is presented to demonstrate the implementation of the proposed approach.

6 ACKNOWLEDGEMENTS

This research was performed at the Institute for Real-Time Computer Systems and Robotics (IPR), Prof. Dr.-Ing. U. Rembold and Prof. Dr.-Ing. R. Dillmann, Computer Science Department, University of Karlsruhe. We would like to thank Prof. Rembold for his support.

7 REFERENCES

AESOP (1994) SIMPLE++ Uebersicht 05/94., Product Description.
8 BIOGRAPHY

Walter Reithofer obtained his diploma in computer science in 1993 from the University of Karlsruhe, Germany. In his master thesis he developed a new micro stereolithography process for rapid prototyping. Prior to his computer science studies, he received a diploma in mechanical engineering in 1987 from the Profession Academy Heidenheim, Germany. Since 1993 he has been a research assistant at the Institute for Real-Time Computer Systems and Robotics at the University of Karlsruhe. His research interests are in computer integrated manufacturing, especially in enterprise modelling and simulation.
Modeling and Reorganizing of Process Chains Using CIMOSA

Barbara Janusz
Institute for Real-Time Computer Systems and Robotics, University of Karlsruhe, D-76128 Karlsruhe, Germany
Phone: ++49 721 608 4264, Fax: ++49 721 606 740,
e-mail: bjanusz@ira.uka.de

Abstract
Enterprise models are used in enterprises for several planning tasks, e.g., redesigning of processes or organizational structures, documentation of processes, planning of the computer usage or software development. For the process redesigning, we use the CIMOSA modeling framework. However, many of the existing CIMOSA models of particular enterprises are not complete, not consistent and not optimal. Also, they often describe only functions or subprocesses limited by department borders of an enterprise.

In order to overcome these problems, we developed a new algorithm. It filters out processes of an existing CIMOSA model of an enterprise. These processes all serve to achieve a particular goal. Using the algorithm, also the completeness and the consistency of the considered processes can be checked. An extension of the algorithm serves for the process optimization.

Keywords
CIMOSA, enterprise modeling, process redesigning, process optimization

1 INTRODUCTION

1.1 Enterprise modeling
In an enterprise, several organizational tasks come up, for example the redesigning of business processes and of organizational structures or the documentation according to ISO 9000. Other examples are planning tasks like planning of the computer usage and the enterprise wide integration of data, optimal usage of resources, the development of control software or the design of new processes.
In order to solve tasks of this kind, a detailed description of the relevant aspects of the enterprise, an enterprise model, is needed. The most important aim of such a model is to provide a detailed and well structured overview of what happens in the enterprise. Using an enterprise model, also inconsistencies in processes can be recognized and their consequences can be analyzed.

1.2 Requirements on enterprise models

Vernadat (1995) presents several criteria which should be satisfied by enterprise models, e.g.:

- **generality and reusability:**
  The model generation is a long and expensive process. Hence, a universal model of an enterprise should exist, which can be reused for each arbitrary task.

- **efficiency:**
  The model should efficiently support the problem solving process, i.e., it should only contain the relevant information for the performance of a particular task. Otherwise, the model becomes too complex and too confused. The efficiency and the generality requirements come into conflict with each other. In this paper, a possibility for solving this conflict is shown.

- **extensibility:**
  Each enterprise model should be able to be expanded.

- **consistency and completeness:**
  These are essential properties of models. It is difficult to guarantee model consistency and completeness, especially in case of multiple users.

- **process orientation:**
  In an enterprise, two components are distinguished: functions (or activities) and processes. The main emphasis of a functional consideration of an enterprise is the description of the task scope of a work place or of a department (Katzy (1995)). Since the model should provide an overview of the enterprise, beside single activities whole processes should be modeled (AMICE (1993)). A process can be partitioned recursively into subprocesses and activities. All subprocesses and activities belonging to a process consume, use or produce objects which are dependent or based on each other. Their execution leads to a common goal (Schaefer (1989)).

- **optimality:**
  The modeled processes should be as straightforward and as optimal as possible.

In the following (section 2), it is shown that the CIMOSA models of particular enterprises do not necessarily satisfy these requirements although the CIMOSA modeling framework provide some concepts which help to achieve this goal. In section 3, an approach for overcoming this problem is presented. Its evaluation is the content of section 4. The paper finishes with an overview on the present and future work concerning this approach.

2 THE MODELING CONCEPTS OF CIMOSA

2.1 CIMOSA modeling framework

This paper is based on the enterprise modeling concept CIMOSA (Open System Architecture for CIM). A comprehensive overview of and a comparison between the CIMOSA concept and other enterprise modeling concepts can be found in CEN (1994) and Naeger (1995). A detailed
description of the modeling framework can be found in AMICE (1993). Therefore, in this paper CIMOSA is introduced only shortly.

CIMOSA consists of an Enterprise Modeling Framework and an Integrating Infrastructure. In the following, only the modeling framework is considered. In this framework, the enterprise model is generated using the following three dimensions:

- Instantiation which is the way to create a model of a particular enterprise using the Generic Building Blocks and Partial Models.
- Derivation which describes the modeling process in three stages: the Requirements Definition Modeling, the Design Specification Modeling, and the Implementation Description Modeling.
- Generation, whereby four views on an enterprise are defined: the Function View, the Information View, the Resource View and the Organization View.

2.2 Modeling according to CIMOSA

The CIMOSA concept provides a user guide for the generation of particular enterprise models (AMICE (1993) and AMICE (1994)). This guide only generally describes the modeling process by defining the steps to be carried out. Therefore, the user does not get a real support from this guide. Some existing CIMOSA enterprise models of particular enterprises, e.g., Zelm (1995), Neuscheler (1995), Hohn (1993), Schlotz (1995) and the CIMOSA user guide were considered and compared with the requirements presented in section 1.2. Following results were obtained:

- generality, reusability and efficiency:
  The models were generated for a specific task, e.g., the design of a logistic system (Schlotz (1995)) or optimization over time in the production (Zelm (1995)). For each other problem, a new model of the enterprise must be generated, because the existing models are strongly limited and hence not universal enough to be reused without an adaptation to the new problem.

- extensibility, completeness and consistency:
  The user guide provides a consistency check at the end of the modeling process without proposing any methods for consistency check. Also, the completeness of a model is not ensured. Therefore, a fault free extensibility method does not exist for CIMOSA models.

- process orientation:
  According to AMICE (1994) and Vernadat (1995) domains must not be confused with organizational departments. They should be logical groupings of processes with the aim to simplify the enterprise. But in the user guide, no methods are proposed which could clarify how to get a process oriented model. Therefore, often, a process is represented by several subprocesses and the connections between them are not obvious immediately. Also, some activities belong to the modeled processes, which lead to other goals than the considered ones.

- optimality:
  In CIMOSA concept, no methods for the optimization of processes are provided.

3 RESTRUCTURING OF CIMOSA MODELS

Summarized, the CIMOSA concept ensures the satisfaction of only few of the requirements presented in section 1.2. The user guide is not strict enough to restrict the user to create a unique model of an enterprise which satisfies the requirements. In principal, some concepts are provided in CIMOSA which should support the meeting of the requirements. But there is no
methodology which could limit the modeling process and force the user to create a model which meets the requirements.

In order to overcome this problem, the following approach is proposed: It is assumed, that a CIMOSA model of an enterprise describing the as-is situation exists. It contains several information about the enterprise. Possibly, the model is not process oriented or the processes are limited by departments borders. Using the following approach, such an arbitrarily structured model can be restructured in order to achieve processes leading to some specific objectives and to better satisfy the requirements on enterprise models. Beside this, an optimal to-be situation is described, which can be realized in the enterprise afterwards. The approach consists of two steps:

1. Reorganizing of the given CIMOSA model in order to get task specific processes.
2. Optimizing the restructured processes.

For the first step, an algorithm has been developed which filters the needed information out of a general CIMOSA model of a particular enterprise (section 3.1). This information forms a process which leads to a given goal. Using the algorithm, incomplete or inconsistent parts of the enterprise model can be detected. In a further step, the reconstructed process can be additionally optimized (section 3.2).

### 3.1 Forming processes

**Modeling activities with CIMOSA**

Using CIMOSA, activities are modeled by the object Enterprise Activity (see Figure 1).

![Figure 1 The CIMOSA object Enterprise Activity.](image)

By definition, an Enterprise Activity transforms its Function Input into the Function Output using its Control and Resource Inputs (AMICE (1994)). Possibly, a Control Output and a Resource Output can be produced. Hence, an Enterprise Activity is defined by its functionality, i.e., its Function, Control and Resource Inputs and Outputs. The Enterprise Activity is a central construct for the algorithm presented in the following. The description of Enterprise Activities on the Requirements Definition Modeling Level can be used, because no resource issues are considered in this paper.

**The process algorithm**

Because the algorithm is used for forming processes which are dedicated to achieve a given goal, the algorithm works backwards from the goal of a process to its first activity:

1. Define the desired output objects of the process.
2. Look for all activities producing these objects as their Function Output. They belong to the last stage of the process.
3. Define the objects which are used by these activities as their Function or Control Input.
4. Look for all activities producing these objects as their Function Output. They are predecessors of the considered activities.
5. Finish, if all currently considered activities do not have any inputs or if they only need inputs which are assumed to be given, otherwise go back to step 3.

The result of the algorithm is a precedence graph which gives the sequence of the execution of activities needed in order to achieve a given goal.

*Example: Forming a process*

*The modeled enterprise:*

In order to show the functionality of the process algorithm, a simplified example is considered. In this example, two departments of the enterprise have already been modeled: the inventory and the production department.

- **The inventory department:**
  The inventory department purchases raw material and receives it (Figure 2). After an inspection of the raw material, its vendor is rated and the decision about using it for manufacturing or reordering it because of material faults is taken.

  ![Inventory](image)

  **Figure 2** The activities of the inventory department.

- **The production department:**
  The production starts with some preparations like getting the raw material and tools and loading the NC program (Figure 3). Afterwards, the raw material is processed in order to manufacture the product. If the raw material is not used completely, the decision about the utilization of the remaining amount must be taken.

  ![Production](image)

  **Figure 3** The activities of the production department.
The process:
The user's interest is the creation of a product. Therefore, he wishes a view on the modeled enterprise, where all needed activities belong to a process "Product creation" and where the irrelevant activities do not occur.

Figure 4 shows the pool of CIMOSA Enterprise Activities describing the enterprise. In order to find the last activity of the restructured process, the algorithm looks for all activities whose Function Output is the manufactured product. In the example, it is the activity "Manufacture". Now, the inputs of this activity must be considered: the predecessor activities must deliver the raw material and the reports on the completion of the tools and machine setups. This procedure repeats until the activity "Purchase" is found, because this activity does not need any inputs. The complete process is shown in Figure 5. All activities in this process participate in achieving the goal "Product creation" and no irrelevant activities belong to it (e.g., "Rate the vendor").

Completeness and consistency of the process
The inputs and outputs of the Enterprise Activities are pointers to other CIMOSA views. Therefore, by guaranteeing the completeness of the Enterprise Activities in the process and by assuming that the inputs and the outputs of an Enterprise Activity are described, the completeness of a process is ensured when all Enterprise Activities belonging to it are described in the model.
Using the algorithm described above, the places in the process where information is missing are indicated to the user. This is always the case, when the algorithm cannot find an Enterprise Activity which produces the Function Output needed as the Function or Control Input by other activities belonging to the process. Hence the user gets detailed indications to information items which are missing.

Concerning the consistency of the modeled information, the algorithm can point out the places, where the consistency of the Function View is not guaranteed. It does not find out inconsistencies in the other views.

Using the algorithm, the user is referred to the points, where inconsistencies can occur. There are several possibilities for inconsistencies:

- different processes which achieve the same goal, i.e., which have the same output,
- different goals for a single process,
- processes with interchangeable Enterprise Activities.

These cases can indicate inconsistencies but they do not do it necessarily. Therefore, the inconsistencies cannot be removed automatically. The decision about possible further steps for removing them must be taken by the user.

### 3.2 Optimization of the processes

In order to form the processes, only the Function and the Information View are considered. Therefore, the result of the algorithm is a precedence graph which gives all possibilities for paralleling and all places, where the activities must be executed sequentially. The Resource and the Organization View are not considered up to now.

It is necessary to investigate the used and the given resources because the feasibility of a parallel execution of activities is only ensured if all needed resources are available. Also the authority and responsibility fields of some organizational units must be considered in order to investigate eventual conflicts while allocating the resources and objects needed for some activities. Beside this, weak points in a process should be detected. Therefore, the optimization of a process is carried out in three steps:

1. feasibility analysis,
2. weak points analysis,
3. evaluation of alternative processes and selection of the optimal process.

Because for these purposes the descriptions of resources and alternative processes are needed, also the Design Specification Modeling Level must be investigated.

#### Feasibility analysis

In the first step, the feasibility analysis, the availability of the needed resources is checked. Up to now, only the static availability for single processes is investigated using the Resource View. The interrelation of several processes at run time is not considered yet.

#### Weak points analysis

The second step, the analysis of possible weak points in a process, consists of several sub tasks (see, e.g., Ferstl (1995)):

1. investigation of breaks in the media chains:
   The media type (readable by machines, by humans or by both of them) should be the same for all information processing activities in the process. Only then problems of multiple data
capture and redundancy, or consistency problems can be avoided. Using the information stored in the Resource View, breaks in media chains can be found. It is the task of the user to decide, how these breaks can be avoided.

2. investigation of changes of the degree of automation:
   For the activities processing physical objects, the degree of automation is investigated. The goal is to get chains with a unique automation degree (fully or partial automated or manual), because only then time, cost and quality losses can be avoided. In analogy to point 1, the Resource View is investigated for this purpose.

3. investigation of the organizational assignments:
   Only if all activities and the appropriate resources and objects belong to one organizational unit, conflicts can be avoided. Then, the coordination expenses and the degree of information exchanges are minimized. Beside this, the organizational unit has an overview of the whole process. In order to detect weak organizational points in a process, the authorities and responsibilities for each activity and the corresponding resources and objects must be investigated. Therefore, for this task all four CIMOSA views are considered.

4. investigation of the assignment of resources:
   For an optimal utilization of resources, they should be released for their reuse as soon as they are no more required for the execution of an activity. Problems can arise if some sets of resources are reserved for the whole duration of an activity and only a part of these resources are needed all the time. In order to find out which resources can be released earlier, the process and the corresponding information stored in the Resource View must be considered.

Beside these possibilities, several other aspects should be investigated. For detailed description of further possible weak points, see, e.g., Ferstl (1995).

**Evaluation of the alternative processes**

Performing the analysis presented in the previous section, the weak points of a process can be detected but they cannot be removed automatically. Therefore, the user decides how he can solve these problems. In most cases, he will develop several alternative solutions, hence his result will be a set of alternative processes. These alternative processes can be evaluated according to several user defined features like time, cost or quality. For this purpose, an evaluation algorithm was developed by Neuscheler (1994) as an extension of the CIMOSA concept. Using this evaluation algorithm, the user can select his optimal process.

### 4 EVALUATION OF THE PROPOSED APPROACH

#### 4.1 Unsolved problems

For both, the process algorithm and the optimization algorithm, some problems must be solved before implementation. They will be shortly presented in the following.

**Process algorithm**

Consider the example in Figure 6 and assume that we look for a process which creates product O3. The result of the algorithm is a set of several possible processes (Figure 7). Difficulties of this kind are due to the only consideration of the inputs and outputs of activities. The semantics of activities and objects and the necessary sequences of the execution of activities cannot be recognized if there are activities with the same inputs and outputs. In order to overcome these
problems, different solutions must be investigated before a solution can be found. Presently, the
three following alternatives are being considered:

![Activity Pool](attachment://activity_pool.png)  ![Precedence Graph](attachment://precedence_graph.png)

**Figure 6** An activity pool. **Figure 7** The precedence graph.

1. marking the processing state of used objects:
   Using particular entries in the object description, it would be easy to recognize the necessary
   sequence of the execution of activities in the process.
2. investigation of the alternative processes:
   In this case, the all alternative processes must be investigated, e.g., processes with loops or
   with different lengths. A solution could be to accept only processes without loops and to
   choose always the longest process for further investigations in order to ensure that all needed
   activities are modeled in the process.
3. classification of the activities:
   For this purpose, a classification similar to the one presented by Vernadat (1995) is investi­
   gated. It is extended by introducing particular properties of function and control inputs and
   outputs which must be set if the corresponding activity is of a given type. Then, additional
   rules depending on these properties and types can be used for the process algorithm.

**Optimization algorithm**

The problem of all optimization algorithms is the semantics. Up to now, only optimization
approaches can be formalized and implemented which are independent of the semantics of
activities or objects. If the semantics cannot be neglected, a method for the classification of
activities like presented in Vernadat (1995) can be used.

### 4.2 The advantages of the approach

With the algorithms presented in section 3, the requirements posed on enterprise models (see
section 1.2) can be fulfilled without forcing the user to create a particular model:

- **generality, reusability and efficiency:**
  An enterprise model describing a wide part of the enterprise can be reused for different tasks.
  For each new task, the appropriate information is filtered out of the model using the
  algorithm presented in section 3.1. The resulted model is as efficient as possible.

- **extensibility, completeness and consistency:**
  New domains or processes can be added to the model. The corresponding Enterprise
  Activities will be considered automatically at the next time the process algorithm is executed.
  The completeness and the consistency are then also checked (section 3.1).

- **process orientation:**
  The result of the process algorithm is a process crossing several department borders. Its
  activities use, consume or produce common information to achieve a common goal.
• optimality:
The modeled processes are simplified, because they only contain these activities and information which are needed for achieving a given goal. The optimization process is supported by the analysis described in section 3.2.

5 CONCLUSIONS AND FUTURE WORK

In this paper, a new approach is presented which helps to satisfy the requirements posed on enterprise models. By filtering task specific processes out of a more general and comprehensive enterprise model, a process oriented model is achieved which only contains the relevant information for the solution of a given task. The processes are optimized using the approach presented in section 3.2. An additional advantage of this approach is, that the user does not need to have a global view on the entire enterprise to generate the models of various processes. Each department or even smaller part of an enterprise can be modeled separately. Only the interfaces between the parts must be clearly defined. The explicit connections between activities in the departments are generated by the process algorithm and thus the process oriented modeling as required in AMICE (1994) is achieved.

Presently, the solution for the difficulties presented in section 4 are being worked out and the approach is being implemented.

Up to now, only single processes are considered. As a next step, the correlation between several processes and a common optimization, e.g., the check of availability of resources at the run time, will be investigated.

Also, possible further applications of the proposed approaches will be investigated. There are different possibilities for the utilization of processes:
• consistency check:
  For the consistency check of a particular enterprise model, processes with the appropriate goals are formed. For ensuring consistency of the originally modeled processes, they must be compared with the formed precedence graph.
• benchmark tests:
  In order to compare several enterprises and their performance, processes with a common goal are formed for each of the enterprises considered. Using the evaluation algorithm presented by Neuscheler (1994) they can be evaluated objectively.
• planning tasks:
  There are several publications, e.g., Naeger (1994) and Schaefer (1989), describing applications for processes. Using the algorithm presented above, these processes can be computed instead of assuming that they are given.

Also the second part of the presented approach, the optimization algorithm, can be used for other purposes. Several ideas exist for extensions of this algorithm in order:
• to implement business process redesigning approaches which are more comprehensive than the algorithm presented in this paper,
• to model organizational restructuring approaches,
• to model some new enterprise paradigms like fractal enterprise (Warnecke (1993)) or holonic enterprise (Seidel (1994)).
6 ACKNOWLEDGMENTS

This research was performed at the Institute for Real-Time Computer Systems and Robotics (IPR), Prof. Dr.-Ing. U. Rembold and Prof. Dr.-Ing. R. Dillmann, Computer Science Department, University of Karlsruhe. We would like to thank Prof. Rembold for his support.

7 REFERENCES


8 THE AUTHORS

Barbara Janusz obtained her diploma in computer science in 1994 from the University of Karlsruhe, Germany. Since January 1995, she has been a research assistant at the Institute for Real-Time Computer Systems and Robotics at the University of Karlsruhe. Her main interests are in enterprise modeling, business process redesigning as well as hierarchical management and control of manufacturing systems.
Integration Frameworks and Architectures
The system integration architecture: A framework for extended enterprise information systems

J. J. Mills and M. Brand
The Agile Aerospace Manufacturing Research Center,
The Automation & Robotics Research Institute,
The University of Texas at Arlington
Fort Worth, TX, USA, 817/272-5927 FAX: 817/272-5952
jmills@arri.uta.edu  mbrand@arri.uta.edu

Abstract

Manufacturing and enterprise formation is changing dramatically with speed and response time key parameters in competitive success. Companies are forming extended or virtual enterprises to tap core competencies and speed up their response times. Unfortunately, current information systems are a major hindrance to the rapid formation of such organizations and often to the rapid response of a single company wanting to change the way it does business. The Systems Integration Architecture (SIA) project is a research project to identify and resolve issues in the rapid integration of dynamic heterogeneous hardware, software and typical in this agile world. SIA is an integration framework based on a new basis model and definition of integration which are briefly described. The background to SIA, some of the issues involved and the approach taken to address them within SIA are presented. Specifically, some of the critical issues in the design of an information infrastructures such as SIA are discussed

Keywords
Systems Integration Architecture, integration, infrastructure, manufacturing, information systems, extended enterprises, virtual enterprises, heterogeneous systems integration

1 INTRODUCTION AND BACKGROUND

Manufacturing is undergoing dramatic and radical changes. Quality has advanced to the point where it is no longer a major competitive advantage and time to market is a key driver. Outsourcing has become a way of life - suppliers are being asked to participate in the design phase of a project instead of simply being handed a set of prints and asked for a low cost bid. Strategic partnerships, with integrated suppliers, qualified on performance, quality and delivery, are now recognized as critical for bringing a low cost, high quality product to market faster than anyone else. Agility, the term coined to describe these changes and paradigm shifts has entered the vocabulary of all manufacturers.
Agility has been defined as the ability to rapidly respond to unanticipated changes in market and customer demands and to thrive and prosper in such an environment (Goldman). In pursuit of “agility”, organizations are reconfiguring their business processes, incorporating “core functions” from suppliers and customers to form “virtual or extended enterprises” in a very short period of time. One goal of these activities is to achieve extremely short response times for product realization and order fulfillment. The environment which these activities create and in which modern information systems must operate includes rapid change and reconfiguration, heterogeneous computer hardware, data bases and applications, and different business rules and control paradigms. Traditional, monolithic systems incorporating a central, shared data base are not designed to accommodate such requirements and new approaches are needed. Indeed, current information systems form a significant barrier to the formation of an extended or virtual enterprise. Integration must take on a new meaning and meet new requirements in this agile, virtual word. Further, agile information systems must facilitate the reconfiguration and the formation of a new business in a very short time, not the 1-2 years required for a conventional approach.

The technology of "integration" itself is complex and vast in scope, from the integration of shop floor motion control devices (Senehi et al., 1991) to the integration of whole enterprises including computers, machines, management organization and human resources (CIM-OSA Reference Architecture Specification, 1990). Integration is a much abused and much misunderstood word that has many meanings and as many implications. For the purposes of this paper, we define the scope of “integration” to encompass the unification of: (a) heterogeneous, distributed computers, (b) heterogeneous data and information, (c) functions provided by software applications (d) diverse control paradigms and business rules, and (e) diverse communications systems. We do not consider the broader issues of how the computers integrate with humans, organizations and other resources. However, under heterogeneous computers we do include any machine that is controlled by a computer.

The Systems Integration Architecture project at the Automation & Robotics Research Institute is focused on deepening our understanding of integration, including reference frameworks, architectures, and information infrastructures for these agile environments, and to explore integration issues within such an environment.

The purpose of this paper is to present our current thinking on the major issues in integrating heterogeneous information systems, discussing various issues and our approaches to addressing them. Specifically, a new basis model for integration is suggested which leads to a new, more appropriate definition of integration. We describe the services required to support this new definition and model of integration and show how the System Integration Architecture (SIA), which is being developed by the Agile Aerospace Manufacturing Research Center, can provide these services. We recommend a standard set of attributes of data interfaces required for plug compatibility and present progress on the design and implementation of SIA.

2 BASIS FOR THE INTEGRATION ARCHITECTURE DESIGN

In previous work, we have identified several issues and presented an approach which addresses them (Mills I, 1995). We describe them briefly below to provide some background
for the discussion of our information infrastructure. The first and major issue concerns the
definition of integration in the environment (scoped out above). Elsewhere, we have examined
previous definitions, models and identified problems with them (Mills II, 1995). Several
features appear common: the concept of a central data base or repository, the need for data
integration to include the ability to manage relationships among data sets (Jarke, 1992,
Wasserman, 1990), the idea of "Levels of Integration " (CIM-OSA Reference Architecture
Specification, 1990), and the need for integration of "processes" or "functions" (Jarke, 1992,
Wasserman, 1990), and control (Nof and Papstravrou, 1992). None of these features address
the problems of trying to integrate dynamic, heterogeneous systems with legacy data and
applications.

2.1 Traditional Basis model

A conclusion we reached during this research was that the traditional basis model (Mills II,
1995) of the logically unified, but physically distributed yet shared data base in a neutral format
is not adequate in an agile, virtual world. It has several flaws, among them: Lack of time
dependence, unification, no concern for the relationships among data sets and the inability to
consider task, function or activity integration.

This traditional basis model has been and still is very useful, and provided a very powerful
concept for developing integrating systems. The importance of a common basis model can be
illustrated by observing what is happening in the object oriented domain with the Object
Management Group using one object model, Microsoft using another, and IBM using yet
another (Betz, 1994). This means that the integration of two systems, based on these two
different object models, is very difficult. A new, more appropriate basis model is needed for
integration in the dynamic agile world of manufacturing.

2.2 Proposed Basis Model of Integration

From the basis model perspective, therefore, the main issue outlined above might be better
stated; "What is the new basis model for integration in an environment of heterogeneous,
physically distributed hardware, operating systems, software and data?"

Elsewhere we have proposed a new basis model based on research in the design theory and
methodology domain (Finger and Dixon, 1989, Nguyen and Rieu, 1987, Takeda et al., 1990,
Suh, 1990, Ulrich and Seering, 1987, Sthanusubramonian et al., 1992, Talukdar and Fenves,
1989). Briefly stated, this model, called the TAR model (Mills II, 1995), suggests that all
information processing simply "transforms" data sets (Figure 1).

The model is intrinsically modular consisting of software and an input and output data set.
The output data set of one becomes the input data set of another (see Section 2.5). Some
transformations may require more than one data set as input. Others may output more than one data set. Transformation modules can also share data sets. The entity which performs the transformations can be as large as a whole business information system (which transforms orders into shipped products and invoices) or modules which transform two numbers into their sum. Modules can be assembled into processes which are simply larger transformation modules (Figure 2). For convenience we call the data sets “Aspects” since they are an aspect of the product model.

Transformations can be manual (i.e. by humans using pen and paper), partially automated (i.e. by humans using some form of computer application to capture the results of his/her transformation), or fully automated (the computer application transforms the Aspect on its own with no human intervention). For brevity, we define the entities which effect transformations as “Functional Transformation Agents” or FTA’s. Three kinds of FTA’s are recognized: Primary, Secondary and Tertiary. Primary FTA’s transform Aspects along the primary product realization process. In the 3-D space formalism of Taylor and Henderson, Primary Transformations transform information sets along the life cycle application axis (Taylor and Henderson, 1994).
Secondary FTA’s move any particular Aspect off this main path into a secondary Aspect necessary to perform some test, verification, evaluation or analysis or to assist a human in some activity necessary to understand a particular product representation. They transform the Aspect into another form within a product realization phase. In Taylor and Henderson formalism, they transform information sets with the Generalization-Specialization levels of detail planes (Taylor and Henderson, 1994). Tertiary FTA’s translate format, see section 2.5.

2.3 New definition of Integration

Integration within this basis model then is the act of defining FTA’s with matching Aspects, and providing mechanisms to support (a) their composition into higher level FTA’s or processes, and (b) their enactment, monitoring and control. If this is done in a permanent manner using a shared data base then we have a traditional information systems solution such as has been implemented in many companies. If we provide an infrastructure in which FTA’s with matching Aspects and modalities can be configured and reconfigured rapidly, then agile manufacturing can be accommodated. Note that this definition is very close to that suggested by Seheni et al. (1991).

2.4 Module Definition

If this basis model and definition are accepted, a concomitant issue is: What should the modules (i.e. FTA’s) be - applications or functions? The traditional, vendor supported view is that since applications are provided by the vendors, the modules should be applications. However, tools or applications are complex combinations of functions which generally (in the case of a suite of applications from a single vendor) cannot be easily separated and can only be readily composed into higher level functions or processes when all the applications to be composed are from a single vendor. From a systems engineering viewpoint, it makes more sense to be able to compose well defined functions with clearly defined interfaces into processes. Examples of “function”: might be “create model,” “edit model,” “assemble product,” “plan production,” etc. Hence, it is our position that the modules (FTA’s) should be based on functions not applications.

2.5 Module Interfaces

The next issue of importance is: “What should the interfaces of the modules look like?” In other words, how should we structure the Aspects so that an Aspect created as the output of one FTA can be the input of another.

In theory, all “data”, including commands to start, stop, pause and resume applications can be regarded as data and transformed. Consequently, one could envisage an interface definition based only on data sets. However, in some domains, a function may require another function (or application) to launch and control a second function. This is important in the domains of shop floor control and virtual manufacturing and engineering. As Seheni et al (1991) have pointed out in their work on manufacturing systems integration, commands need to be separated from data, and they, in fact, identify five different components to an interface. To simplify matters, only two major interface components are recognized: (a) administrative
commands such as start stop, etc., and (b) Aspects, the data sets in the TAR basis model. Figure 2 illustrates the concept of one FTA commanding another by means of the thicker arrow.

The purpose of an Aspect is to be able to match up the output Aspect(s) from one or more FTA's to the input Aspect(s) of other FTA's so that they can be formed into processes. In the System Integration Architecture project at the Automation & Robotics Research Institute, research has indicated at least six attributes are required to define the structure of the Aspect so that this matching can be achieved reliably: the data model or type of data, the data format (e.g. DXF, IGES, STEP for CAD, RTF, ASCII for text, TIFF, HGL, PCX for graphics, etc.) the existence, the location, the physical storage structure (e.g. file or database) and the access type (e.g. read/write, database query, STEP SDAI, RPC, API or http call) of the actual data set. These attributes are not unrelated. Clearly if the data are accessed through an API, then the physical storage attribute is not relevant. This structure incorporates the idea of the data model and the Standard Data Access Interchange (SDAI) of STEP, but broadens the concepts. It is proposed that this structure be called the Modality of the Aspect since different values of the attributes describe different modes of the Aspect.

A two step method of use for a Modality has been identified. During process composition (i.e. build time) only the data model attribute need be matched to form a process. At run time, however, the data format, existence, location, physical storage structure and access type become important for an executable process. Figure 2 illustrates the concept of linking FTA's through Aspects, into processes. It also illustrates the concept of Tertiary FTA's which simply transform the modality of an Aspect. These are typically the translators provided by CAD vendors.

3 DESIGN ISSUES IN THE SYSTEMS INTEGRATION ARCHITECTURE

The next issue is: “How do we support rapid reconfiguration of FTA's?” That is: What kind of infrastructure is required to support the creation, enactment and control of FTA's? What services should this product provide for legacy systems, heterogeneous computers, functions and data? The System Integration Architecture (SIA) will provide the services necessary to address this and other, related issues. SIA is based on an object oriented, client server paradigm. We are currently using Iona Technologies’ Orbix implementation of the Common Object Request Broker Architecture (CORBA) to facilitate distributed objects for our client-server environment.

3.1 Access to Remote Functionality

The infrastructure must communicate with an FTA whose application resides on one remote computer while its Aspect(s) are located on another computer. The infrastructure enacts, monitors, and controls the FTA execution process. The term “execution process” indicates that an application has been enacted and remains within a computer's execution space. FTA’s must also have access to Aspects which reside on other computers. Finally the
The system integration architecture

The system must allow FTA's to be enacted in a predefined order. These requirements for the infrastructure are discussed below with some of the design and implementation details of SIA. While this paper is focused on manufacturing, the issues addressed in this project are much more general and can be applied to a wide variety of other domains.

The Launching Server: The infrastructure must communicate with each node containing an Aspect or an application which is a part of an FTA's definition. A server is installed on each computer within the SIA network to provide launching capabilities. The launching server receives incoming communications and provides local functionality for controlling FTA executing processes. The launching server must enact new execution processes, provide monitoring functionality, and allows an end user or another FTA to control the execution process by sending commands to kill, pause, and resume the process.

When the launching server receives a message to enact a new FTA it provides services for notifying the end user if the execution process sends standard output or error messages or if the process terminates normally or otherwise.

FTA's must be spawned as a new execution process so that the launching server's can respond to new incoming messages. Multiple execution processes can therefore be monitored by the same launching server. We are investigating possible solutions to this problem. One possibility is for the dispatcher to be enacted as a forked (multi-threaded) process. A second execution process, or thread, is created which in turn enacts the FTA's execution process. The forked thread can monitor the output streams and will terminate when the FTA process terminates. The forked thread must communicate with the infrastructure to provide notification of errors or termination.

We have encountered a significant degree of complexity while implementing the launching server. Maintaining communications between the parent and children forked processes has proven difficult using the normal Orbix implementation. The multi-threaded version of Orbix would add a considerable degree of complexity and cost to our implementation. Because launching servers represent the most important limit on the scalability of SIA we are investigating other possibilities.

Distributed Communications: The infrastructure must provide communications between the network computers. The communications module must provide distributed communications between the various launching servers. Several methods of distributed communications are available as third-party products. We use third party products wherever possible within SIA to maximize re-use and efficiency. The Communications module must provide distributed communications protocols which are efficient, integrate easily, accessibility, and cost.

We initially implemented distributed communications for SIA using TCP/IP and UDP communications protocols. We converted to Orbix as a result of the emergence of Object Management Group's CORBA as a standard for distributed object communications ("The Common Object Request Broker Architecture and Specification", 1993). CORBA provides distributed communications between objects regardless of which execution space or computer they reside. However, CORBA must be installed on each machine that provides a launching
server or end user interaction. Installation of the vendor product requires both a time and economic investment for each computer and so reduces scalability.

RPC and HTTP are communications protocols which are commonly installed on networked computers. RPC is readily available and has existed for long enough to be considered a stable protocol. RPC is not, however, as directly applicable to the object paradigm. HTTP also appears to have much promise. The popularity of the World Wide Web, which uses the HTT protocol, suggests that in the near future most, if not all computers, will have an HTTP installed. The World Wide Web’s dependence on HTTP servers ensure that they will be freely available and easily accessible for all platforms. HTTP also provides the ability to transfer data and execute processes on remote network nodes. HTTP was not, however, designed for inter-process distributed communications and so is not as robust as CORBA or RPC.

It is not clear which, if any, protocol will emerge as the dominant standard. Accordingly, we are implementing SIA so that distributed communications protocols themselves are rapidly reconfigurable. We can then compare the different methods of distributed communications and choose the most applicable. The design will also allow for dynamic reconfigurability of the communications protocol. Each node may have some different form of communications previously installed. Dynamically reconfigurable communications protocols allow new computers to be integrated rapidly using their native communications protocols.

**Distributed Aspects (Data Sets):** FTA’s generally require input and output Aspects. Aspects are composed of data stored somewhere on the network. The FTA’s application must have access to an input Aspect and must provide output Aspects to the infrastructure. The infrastructure must, therefore, transfer or otherwise make available data which exists on one network computer to an application which executes on another.

The infrastructure must provide methods of transferring data securely through the network on demand. The infrastructure must also know the Modality (including the location) of the input Aspects before an FTA can be enacted. It may be necessary to extract data from within a database for use as input to the FTA. A file may be transferred from one computer to another, but the copied file is considered to be transitory and will be destroyed upon completion of the FTA’s execution process. Output Aspects must be dealt with in a similar manner. If an output Aspect is defined to reside on a network computer other than that of the FTA’s application, the Aspect file must be transferred to the specified computer. Only one copy of a particular file should exist on the network in the location specified by the FTA design unless a transitory Aspect is in use.

**Definition of Complex FTA’s:** We define the composition of a number of FTA’s and their Aspects into a process, a *Complex FTA*. The Complex FTA provides a higher level transformation. The implementation of Complex FTA’s is another important issue for the infrastructure. A *Simple FTA* is defined as an FTA which enacts a single execution process. A Complex FTA links multiple simple FTA’s together so they can be executed in a predefined order. Complex FTA paths are defined by a directed graph data structure and must allow parallel paths and iterations. Multiple parallel processes can execute using a target-dependent structure to maintain a proper order of execution. Fig 2 shows an example of a complex FTA graph.
A Complex FTA is similar to a Simple FTA in that it transforms a set of input aspects to a set of output aspects. A node within a complex FTA directed graph can be either a Simple or a Complex FTA. An infinite degree of scalability is therefore available within FTA design. The designer of Complex FTA need not know or understand the internal structure of another complex FTA to use that FTA within the new design.

There are two directed graphs which represent any Complex FTA. The first is the highest level graph which shows a directed graph in which each node is either a Complex or Simple FTA. Figure 2 shows the high level Complex FTA graph. The second graph is an exploded view of the high level Complex FTA graph in which each node in the graph represents a Simple FTA. Figure 3 shows the exploded Complex FTA graph version of Figure 2 with the exploded complex FTA highlighted in grey. In this figure we have dropped the data base representation of the Aspect in the Complex FTA for clarity.

3.2 Entity Design

SIA is an information infrastructure which locates and communicates with network services. The infrastructure must also address the issue of identification and storage of FTAs and other information. The information required to identify, launch, and control FTA’s is stored within a second key SIA module, the Librarian. The Librarian is the central point for the infrastructure and usually resides on a single computer. The Librarian provides a database which stores centralized information. A special type of object within SIA’s object oriented architecture is called an entity or reference. The objects provide information about applications, aspects, and FTA’s. The entities also allow information to be accessed and for messages to be sent from the entity to other network nodes. Entities can also store administrative information such as User information.

**Identification of FTA’s, Aspects, Applications, Functions, and Jobs:** FTA’s, Aspects, and applications are all network services for which specific entity classes must be defined. An FTA entity identifies the application, as well as input and output Aspects associated with the FTA. Aspect entities record the Modality of the Aspect. Application entities record the location of the executable file.

Many applications can perform multiple tasks. The specific task performed is often determined by command line parameters or input data. A function is defined within SIA as the

![Complex FTA Diagram](image-url)
execution of a given application in a specific mode to perform a specific task. FTA’s can be defined for each function rather than just each application. A critical distinction between applications and FTA’s is that the FTA entity also includes information about the Aspect(s) required for that function.

Information about execution processes resulting from launching an FTA must also be maintained. The Job entity is introduced to provide this service. A “Job” is created when the end user requests that an FTA be launched. At this point the end user may be called upon to identify any Aspects whose physical names and locations were not predefined within the FTA’s definition. After the Job is created and the Aspect locations are identified, SIA sends a launching request message to the appropriate launching server.

**Administrating Access to FTA’s:** As networks increase in size the number of network services (i.e. FTA’s) available to an end user will increase geometrically. Scalability is therefore an important issue for an agile information infrastructure. The number of available FTA’s must be filtered to insure that the end user can easily find FTA’s to address a particular problem. SIA introduces administrative entities to address these issues. Like other entities, administrative entities provide storage of relevant data and the ability for other entities to communicate among themselves and network services.

We have identified several administrative entities which can help reduce the complexity of an infinitely scalable network. These include User, Project, and Group. User entities identify users so that the set of available information is reduced to only that required by a particular User. Projects allow FTA’s, Aspects, and Users necessary to address a particular problem to be grouped together. Group is similar to but more general than the Project Entity.

Security will also be an issue for a large distributed system. Only a small number of Users should be allowed access to a given project. Administrative entities allow access to Projects, Groups, Aspects, and FTA’s to be limited to the subset of Users.

### 3.3 Access Control

**The Graphical User Interface:** SIA has been implemented in modular fashion to maximize the efficiency of network communications. The Librarian is a central server existing on one network computer and providing persistent storage of SIA’s entities. An end user accesses this database by enacting SIA’s graphical user interface (GUI) on a local machine. Once logged in, the user is presented with a list of projects which the end user may address. A list of jobs which the user has launched is also available, allowing them to monitor and control the remote execution processes associated with a launched FTA. The end user can select a project, at which time a list of appropriate FTA’s will appear. The end user can choose to launch an FTA from this point.

The end user can exit SIA’s GUI at any time without affecting launched processes. If an execution process sends standard output or error messages and the user launching the job is running a GUI, these messages are sent immediately to that user’s monitor. If there is no GUI open for the user, the Librarian will simply store them. When the user next enacts a SIA GUI, the messages will be immediately sent to that monitor. This allows a user to terminate the SIA.
GUI during long executions and return later without losing any information regarding the execution or the results.

3.4 Design Flexibility

SIA Modules and Layers: Each “module” contains internal layers which provide specific services. Layers generally communicate with only layers directly above or directly below them in the architecture. The layers are also easily replaceable by alternate layers having different implementations. Figure 4 presents an overview of the current design.

The Librarian Module: The Librarian module the central module of SIA. All inter-modular communications pass through the Librarian to determine the network location of the message recipient. A database is located within the Librarian module to persistently record the message. The Librarian has administrative and functional components which provide access for those services to the database. The database forms the final layer within the Librarian, providing passive data, but no active services.

The Executive Module: The Executive module provides the end user access to SIA through a graphical user interface layer. The Librarian is a server for the Executive providing it with requested entities. The Librarian can also be a client of the Executive, notifying the Executive of any change in the state of launched jobs. The Executive module includes an entity “buffer” layer which provides local access to the remote Librarian entities upon request, and a control subsystem to control the execution of complex FTA’s. There can be many executive modules distributed across the network.

The Communications Module: The Executive and Librarian modules need not reside on the same machine. The Communications module provides access between the Executive and Librarian and among the other modules. Executive, Librarian, and Dispatcher Communications layers are defined within the Communications module to provide distributed communications between the Executive and Librarian modules and the Librarian and Function modules.
The Function Module: The Function module includes the set of all FTA's, launching servers, dispatchers, remote processes, and remote data available to SIA. Launching servers and dispatcher are being developed for UNIX, PC, and Macintosh operating systems. There are many Function modules spread across the network.

4 SUMMARY AND CONCLUSIONS

Various issues in the integration of information systems in an agile environment have been discussed and the problems with the traditional basis model or abstraction presented. A new basis model and definition of integration are proposed and the services required for an integrating system suggested. A brief overview of the System Integration Architecture project shows how the Agile Aerospace Manufacturing Research Center is attempting to address these issues by implementing a new architecture based on the proposed basis model.

The Systems Integration Architecture has been implemented as a framework which is easily extensible for any project that requires an information infrastructure framework. We have also designed SIA to have the flexibility to experiment with alternate technologies whose introduction may contribute to the efficiency of some portion of the project. SIA is being implemented, not as a product which is meant for market, but as a research framework which can be modified and tuned for various needs. The modularity of the design including the reconfigurability of the communications and database layers provides many advantages for research into integration issues in the dynamically changing domain of agile manufacturing.

5 ACKNOWLEDGMENTS

The author acknowledges the financial support of the National Science Foundation from cooperative agreement DDM-9320949. This work is being performed as part of the Agile Aerospace Manufacturing Research Center. The assistance of Dr. Ramez Elmasri in the data base area is acknowledged. The contributions of Dr. Marc Turcotte in the design and implementation of SIA is gratefully recognized.

6 REFERENCES


Dr. John J. Mills received his Ph. D degree in Applied Physics at the University of Durham, Durham, England in June 1965. As the Director of the Automation & Robotics Research Institute, Dr. Mills manages a multi-million institute which is dedicated to helping US industry become globally competitive. Dr. Mills is also the director of the Agile Aerospace Manufacturing Center, one of three Agile Manufacturing Institutes in the nation. Funded by an ARPA/NSF Agility Initiative, the Center is helping to define agility in aerospace manufacturing in the United States. He also serves as a member of the Board of Directors for several organizations including the ASME Manufacturing Technologies Group Operating Board (MTGOB), the Agile Enterprise Quarterly Editorial Board, the Academic Coalition for Intelligent Manufacturing Systems (A-CIMS) Board of Directors and others.

In addition to his administrative responsibilities, Dr. Mills also maintains an active research program focusing on the information infrastructures required for agile manufacturing. He is author of over 100 publications including book chapters, articles in prestigious journal, and industry reports.

Mike Brand has been the project manager for the $250,000 Systems Integration Architecture project of the Agile Aerospace Manufacturing Research Center at the Automation and Robotics Research Institute of the University of Texas at Arlington since September of 1995. He is also the chief architect for the Systems Integration Architecture, producing all high-level analysis and design and directly supervising all lower level analysis and design. Design issues include identifying client-server partitions within the architecture, implementing architectural design changes for the development of reconfigurable architectural layers, supervising the implementation of distributed object client-server applications using the CORBA compliant Orbix software from Iona Technologies, and supervising the implementation of an object oriented database using Object Design’s Object Store. Mike Brand graduated with a Bachelor of Arts degree in Physics from Earlham College in Richmond Indiana and expects to receive a doctorate in Physics from the University of Texas at Arlington in May of 1997. His graduate work involved designing and implementing an application capable of processing and rendering data about solid state Fermi surfaces retrieved from a two-dimensional angular correlation of annihilation radiation (2D ACAD) experiment.
A Model Based Approach to Enterprise Wide Information Support

P.E. Clements, I.A. Coutts and J.M. Edwards
MSI Research Institute,
Loughborough University, Loughborough, Leics, LE11 3TU, UK.
Tel: +44 1509 228250
Fax: +44 1509 267725
Email: p.e.clements@lboro.ac.uk

Abstract
This paper demonstrates an ability to bridge between conceptual information models representing a manufacturing enterprise and the realisation of these models within a relational database management system. The information architecture defined within the paper is based on the ANSI Three-Schema approach to database management which splits the data management issues into three separate parts and thereby realises important benefits in complex systems where information sharing needs change frequently.

The tools created to implement this information architecture support the following lifecycle phases of requirements definition, detailed design, and system execution and maintenance. This paper concentrates on the implementation and system execution issues.

The approach provides direct support of system change by providing structured diagramming and mapping tools which semi-automate information requirements definition (and ongoing change in requirements), and model driven systems implementation tools which facilitate the construction and runtime management of the required system using the services of an integration infrastructure.
1 INTRODUCTION

(Sun Microsystems, 1994), in their DOE project, stated that:

"Modern-companies are increasingly becoming information based organizations dependent upon the continuous flow of data for virtually every aspect of their operations. However, their ability to handle data is breaking down because of the volume of information is growing at a faster rate than their ability to process it. Computer hardware performance is not the problem. The last decade has seen a ten-thousand fold increase in processing performance. The failure lies in software: traditional approaches to designing, developing, and maintaining information processing systems continue to limit our ability to manage data. The lack of parity between hardware and software systems along with the flexibility limitations of monolithic information architectures waste computer resources. Everyone is affected: the company, its projects, and users. The problem is not just with the developer or development process, but with the underlying technology upon which the information systems are built. To move forward and escape the limits of traditional information processing methodology requires a better approach -- one which employs a more flexible paradigm for the construction and maintenance of software systems so that they can meet the ever-changing needs of companies today."

This quotation supports a general consensus view that a key issue of the 1990’s is the ability for world class manufacturing enterprises to handle their information in a structured and controlled manner. Ad-hoc approaches to the design and maintenance of information systems need to be replaced by a more controlled and flexible ones. In addition an ability to readily respond to changes in information needs of the enterprise is required. However any new approach should maintain a closer correspondence between organisational procedures, adopted to support the design, construction and maintenance of information systems, and their underlying supporting information structures.

The Manufacturing Systems Integration (MSI) Research Institute have, over a number of years, developed such an approach through seeking ways of bridging between the theoretical world of system models and the physical world of implementation technology. See Figure 1.

The methodology proposed and described in this paper seeks to meet the following requirements:

(a) that enterprises must react in a flexible and speedy manner to their changing information requirements;
(b) that end-users must be disassociated from implementation details;
(c) that user applications must be isolated from implications of implementation technology.
Enterprise wide information support

Theoretical Modelling and Analysis of CIM Systems: Supporting their Conceptual Design and Analysis

"Void" or "Gap" between the two worlds, except for the existence of limited scope point solutions which link particular views or aspects of an integrated system

Bottom-up Implementation techniques and RunTime processes: Supporting the building, execution and management of a physical system

Enable formalism and problem decomposition, but can’t adequately address implementation issues and constraints

Enable the creation of working systems, but will not facilitate optimised design (particularly where systems of wide scope and high complexity are involved)

Figure 1. Need to bridge the “Gap” between Theoretical and Physical Worlds.

The paper is structured within three principal sections which describe a) the approach, b) the method of implementation and c) a case study which demonstrates the benefits of the methodology.

2 THE APPROACH: A MODEL-DRIVEN INFORMATION ARCHITECTURE

Researchers at the MSI Institute have sought to bridge between models of candidate systems and real world physical implementations of them in various ways (Aguiar, 1993), (Gilders, 1995), (Murgatroyd, 1993) by considering different perspectives (primarily function, behaviour, resource and information views) of an enterprise. This paper concentrates on using models to drive the information architecture of an enterprise in order that the following can be achieved:

(a) separation of the user/application developer from a need to understand particular characteristic properties of the underlying database system in terms of structure, mechanisms and management paradigms
(b) operation in an heterogeneous database environment;
(c) accommodation of legacy based information systems, and;
(d) support for system change

MSI adopted the ANSI Three-Schema approach, as a means of building a model based architecture. The ANSI approach defines three schemas or views of how database systems should be managed, namely:

(a) the External schema, which is the users view of the database;
the Conceptual schema, which is an abstract definition of the database, and should represent the data and relationships between the data without considering the physical resources available or the database system within which it is stored;
(c) the Internal schema, which deals with the physical representation of the data and its organisation.

The relationship between the schemas is shown in Figure 2,

![Figure 2. The ANSI Three Schema Architecture.](image)

Based on a Three-Schema approach MSI researchers developed an information architecture which covers the various phases in the lifetime of information systems, see Figure 3. This figure demonstrates a model driven approach to lifecycle support. It also introduces the principal information tools which were produced to realise model driven database creation, population and runtime access.

The following sections of this paper detail the manner in which the three schema are supported via an integrated set of tools.

2.1 Design Phase

During the design phase of an information systems project a description of an information model is created in a neutral format from which a number of database implementations can be built. This neutral model corresponds to the conceptual information schema.

After an initial survey of the techniques available for describing a conceptual schema a decision was taken to use EXPRESS and EXPRESS-G. This decision was based on the following principal issues: (a) EXPRESS-G provides a good graphical interface to the EXPRESS language (b) EXPRESS is readily computer readable (c) within the manufacturing domain EXPRESS had already been adopted by the STEP community (ISO, 1995) and (d) at the time of the decision in 1991, there was a strong possibility of it being adopted as an ISO standard.
One major problem was that in 1991 when the research reported in this paper began, there were no tools available to transform an EXPRESS model into a set of relational tables. Egg- ers, 1988 had proposed an initial mapping of the EXPRESS constructs onto the relational paradigm but no actual tools were available. Hence, the first stage of implementing the conceptual schema involved creating a tool to transform an EXPRESS model into a relational form (Clements, 1991) and in particular be consistent with requirements of the INGRES database. An EXPRESS-to-SQL compiler was written which generates as outputs: (a) a set of create table statements which correspond to the EXPRESS model, and (b) a set of files which allow the EXPRESS model to be reconstructed for subsequent use during implementation and execution life-phases.

2.2 Implementation Phase

Once the structure of the database has been created the next logical stage was to develop a method populating data repositories.

As part of the STEP standardisation work which occurred alongside effort aimed at realising the EXPRESS language description, the STEP standard sought to define a physical file
description which allows a data repository defined by the conceptual model to be populated by an ASCII file. As for the conceptual schema in 1991, no tools were available hence a STEP parser was created which:

(a) checked the STEP physical file for syntactic correctness according to the proposed standard;
(b) checked the STEP physical file for semantic correctness against the particular EXPRESS model being populated, and;
(c) created the insert table statements required to populate the data repository.

2.3 Execution Phase

The main design criteria for the execution phase of the information architecture is the ability to separate logical descriptions of information from their physical implementation. Hence the information architecture contains a tool which allows application developers to specify a logical name which maps onto a set of attributes specified by the conceptual model. Thus at runtime an application or end-user can execute the logical objects, and mechanisms within the information architecture will map the logical name onto the appropriate EXPRESS attributes and thereby automatically create SQL statements. These statements can then be executed by engines provided by the integration infrastructure, and presented back to the end-user/application on successful completion of the database request.

3 THE IMPLEMENTATION: AN EXECUTION ENVIRONMENT

The mechanisms described in the following sections are concerned primarily with the runtime execution environment. This environment, see Figure 4 was created to demonstrate the model driven Three schema approach to information integration.

3.1 Implementation of the External Schema

When a user application needs to access particular information contained within the underlying databases it can realise this via a number of services. For each of these services the user needs to pass a limited set of identifiers (the specifics of which will depending upon their security profile) to the information view provider, Eventually the user application will eventually receive data back from the view provider in a format previously specified by the user/user application.

The following services are provided in the current implementation:

(a) obtain - this allows the user to receive information concerning the particular identifier passed to the view provider;
(b) submit - this allows the user to add information to the underlying database;
(c) modify - this allows the user to modify particular information within the database;
(d) erase - this allows the user to delete a particular item from the database;
3.2 Implementation of the Conceptual Schema

On receipt of an information request, the view provider has to perform a number of tasks, namely:

(a) map the logical name onto respective attributes identified by the EXPRESS models. For example, CARwheel has the attributes radius, make, frontOrBack;
(b) generate a set of SQL statements which collectively service the requested call. This function is enabled by storing an internal representation of the EXPRESS models in the view provider environment (set of tree structures) from which it can deduce the set of necessary statements;
(c) pass this set of statements to relevant device drivers for submission to their respective local database management systems; and
(d) receive the results of the query and format the results in which is meaningful to the requesting application.

3.3 Implementation of the Internal Schema

The integration infrastructure provides the location transparency required to implement the internal schema. It does this by providing the services necessary to enable interaction between various software objects (viz: user applications and the view provider application) which com-
prise a system. It provides a consistent set of services which is independent of the object’s physical location, or the operating system and network protocols used. Researchers at MSI created such an integration infrastructure called CIM-BIOSYS (CIM Building Integrated Open SYStems) in the late 80’s, a detailed treatment of which is given in (Coutts, 1992).

A message packet generated by the view provider contains information regarding necessary SQL requests to a database at some “logical” location. Using a method analogous to the view provider approach, CIM-BIOSYS uses configuration data to identify the physical location of the database and thereby directs messages to appropriate drivers. The driver interprets the request, interrogates the database and creates reply messages sending them via CIM-BIOSYS to the view provider.

4 AN INDUSTRY BASED CASE STUDY

To evaluate and demonstrate the benefits of MSI’s approach to information systems design and construction a proof of concept case study system has been produced which is based on plant systems and data currently in production at the site of a UK contract manufacturer of high complexity printed circuit boards (PCB’s). The application area chosen was the control of a surface mount PCB assembly line, see Figure 5.

The demonstrator system comprises a number of user applications which supervise, control and monitor the “printing” of PCB’s with solder paste, the “population” of PCB’s with components and the “inspection” of PCB’s following placement and reflow soldering. A single information view provision application handles the requirements of all user applications in the case study system.

4.1 External Schema of the Demonstrator System

An EXPRESS model was created which structures the information entities and entity relationships of the demonstrator system, see Figure 6.
SCHEMA execution;

ENTITY assemblyLine;
    name : STRING;
    machines : SET[0:?] OF machine;
    operators : SET[0:?] OF operator;
    batch : batch;
    printInspRate : INTEGER;
END_ENTITY;

ENTITY machine ABSTRACT SUPERTYPE OF (ONEOF (printer, conveyor));
    currentBoard : board;
    name : STRING;
END_ENTITY;

ENTITY operator;
    name : STRING;
END_ENTITY;

ENTITY printer SUBTYPE OF (machine);
    pasteHeight : REAL
END_ENTITY;

ENTITY conveyor SUBTYPE OF (machine);
    speed : INTEGER;
END_ENTITY;

ENTITY batch;
    boards : SET[0:?] OF board;
    noOfBoards : INTEGER;
END_ENTITY;

ENTITY board;
    quality : STRING;
    status : STRING;
    id : INTEGER;
END_ENTITY;

END_SCHEMA;

Figure 6. An abridged version of the EXPRESS file.

Then a number of information identifiers were created and assigned to information entities to allow user applications to interrogate the data repositories without needing to know the underlying database structures. In this case study an Ingres database was used to store the physical data. Typical identifiers included PrinterInspectionRate and StatusOfBoardsOnLine.

4.2 Conceptual Schema of the Demonstrator System

For this example two information identifiers were specified, PrinterInspectionRate and StatusOfBoardsOnLine which in the conceptual schema are mapped as below:

(a) PrinterInspectionRate -> execution.assemblyLine.printInspRate;
(b) StatusOfBoardsOnLine -> execution.assemblyLine.name, execution.board.status.
On receipt of an information request from a user application (which is transmitted via the integration infrastructure), the view provider makes use of the EXPRESS model to create the set of SQL statements required to execute the query, see Figure 7. The SQL statements are then packaged in a message and submitted to the integration infrastructure which transmits them to the Ingres device driver so that the queries can be serviced by the database.

Subsequently the view provider will receive an incoming message from the integration infrastructure which contains the results of the query. The results are then transformed into a predefined presentation format and transmitted to the requesting application via the integration infrastructure.

4.3 Internal Schema of the Demonstrator System

To enable the Ingres database to be accessed via standard CIM-BIOSYS services, a device driver (Leech, 1993) was created which essentially enables CIM-BIOSYS and Ingres to communicate in a meaningful manner. The basic functionality of the Ingres driver was to:

(a) create a physical link using UNIX pipes between CIM-BIOSYS and the Ingres database management system;
(b) initiate a database session for CIM-BIOSYS within the Ingres database;
(c) submit SQL statements generated by the view provider to the Ingres database, and;
(d) pass the results of the queries back to the view provider.

Figure 8 shows the internal schema of the Ingres database, that represents the EXPRESS model shown in Figure 6 and which was created automatically from the EXPRESS model by the compiler. It can be seen by examining Figure 8 and Figure 6 that the compiler creates intermediate tables (for example e4) which are used in join statements to link the entity “assemblyLine” to the entity “batch”.

5 CONCLUSIONS

The approach described in this paper provides support for two classes of system change, (a) the system information requirements through its model driven approach, and (b) the system implementation technology through its deployment on an integration infrastructure.

More specifically the approach addresses the three requirements identified in the introduction:
(a) enables enterprises to react in a flexible and speedy manner to their changing information requirements through enabling tool supported creation of information elements which automates much of the system programming which has always been the cause of delay in information systems in the past.;

(b) it decouples end-users from implementation details by realising a logical to physical mapping which associates user information requests to database fields;

(c) it realises a level of implementation independence through its use of an integration infrastructure. The use of CIM-BIOSYS meant that the methodology development could be focused on information design, creation and execution whilst the integrating infrastructure allowed location transparency and provided configurable connection to external packages.
6 ACKNOWLEDGEMENTS

The authors wish to express their thanks to the members of MSI who aided in the development of this methodology and in particular Marcos Aguiar, Jack Gascoigne, Paul Gilders, Allan Hodgson, Shaun Murgatroyd and Richard Weston. Also thanks to the CDP directorate of EPSRC for their continued support of the work.

7 REFERENCES


Coutts, I. A. et al. (1992.) Open Applications within Soft Integrated Manufacturing Systems, Proc. of Int. Conf. on Manufacturing Automation, Hong Kong, ICMA


The Glue Logic: An Integrated Programming/Execution Environment for Distributed Manufacturing Work-Cell Control System

M. Takata  
The University of Electro-Communications  
1-5-1, Chofugaoka, Chofu, Tokyo 182, Japan, Tel: +81-424-83-2161 ext.4744, Fax: +81-424-84-1598, email:takata@cc.uec.ac.jp

E. Arai  
Osaka University  
2-1, Yamada-oka, Suita, Osaka 565, Japan, Tel: +81-6-879-7555, Fax: +81-6-879-7570, email: arai@mapse.eng.osaka-u.ac.jp

Abstract
In this paper, a infrastructural system designed for the factory automation applications, named “Glue Logic”, is described. The Glue Logic provides an environment which supports the programming, controlling and monitoring the manufacturing system, mainly in the levels of manufacturing work-cells. And this environment also supports efficient manufacturing programming, by means of high program modularity and reusability.

Using the service of the Glue Logic, users can easily implement data-sharing and task-interlocking among multiple application processes developed and compiled separately. Furthermore, this system includes event notification message sending and condition monitoring features to eliminate needs of data polling by means of active database technique.

As the result of the use of the Glue Logic, the application system can be build as a collection of the agents working together, and the Glue Logic itself acts as a message exchange and shared data space manager.

Keywords
Factory Automation, Manufacturing work-cell control system, Infrastructural Software System, Distributed Programming / Execution Environment
1 INTRODUCTION

The Glue Logic is an infrastructural system which is designed to make building manufacturing work-cell control systems easy and flexible (Takata 1995) (Takata 1993). This system binds multiple application software modules developed and compiled separately, and coordinates those modules by means of inter-process massage passing.

As the Glue Logic supports event notification and condition monitoring features based on active database scheme, users can easily build event-driven application programs. Each program modules are free from polling shared data, waiting for notification messages.

All the data and application modules in a system can be represented by symbolic names defined in the Glue Logic, and are accessed without any knowledge about implementation of other modules. Each modules in an application system can be developed concurrently, and can be added, deleted or changed freely without modifying other existing modules.

As the result, Glue Logic compliant software modules are easy to re-use, and the users can build large libraries of application software modules. Furthermore, the life cycle and the reliability of such modules are extended, and their development cost is greatly reduced.

This paper describes the Glue Logic designed as the infrastructural system for factory automation applications, and is now under prototyping. In Section 2, the characteristics of the manufacturing work-cell control softwares are discussed. Section 3 discusses the description of the Glue Logic, and lastly, Section 4 discusses on the design of a coming global programming language, to which the Glue Logic provides its execution environment.

2 TARGET OF THE GLUE LOGIC

2.1 Target system controlled

The Glue Logic is designed for controlling manufacturing work-cells, which are operated in the flexible floor shop manufacturing systems. The manufacturing work-cell considered here consists of a work-cell control system and devices such as NC machines, assemble machines, robots, conveyers, storage systems, sensors, actuators, and so on.

Currently, there are many micro-processors in a manufacturing work-cell, which are used within FA controllers, NC controllers, robot controllers and Programmable Logic Controllers. But the rapid progress of recent technology implements low-cost high-performance micro-processors, and also implements the possibility to have integrated manufacturing work-cell controllers, which controloes NC's, robots and DI/DO directly.

In order to operate the integrated manufacturing work-cell controllers, the software systems should have steady foundations, that is the execution environments. The aim of designing the Glue Logic is to provide an environment described above.

2.2 Requirements for the Glue Logic

Based on the observation of the current manufacturing work-cell control software systems, the manufacturing work-cell control software systems and its support system should have following characteristics:
Manufacturing control software systems should have sense of time.
- As multiple processes run concurrently, abilities to control other processes are required. For the maintenance, those operation should be denoted concisely in program texts.
- Exception handling and the execution flow control of the exception handlers are very important for the application.
- Abilities to exchange information among multiple processes running on multiple work-cell control systems are vital for the software.
- Abilities of information sharing and keeping information consistent are required.
- Program modules are frequently added, deleted and altered, even in the manufacturing line is under operation.

In order to ease programming application systems, the following system design paradigms are strongly recommended:

- Building manufacturing control program system as the collection of modules.
- An infrastructural system should be introduced to bind application program modules, and each module should be designed to be the infrastructural system compliant.
- The cell models can be build in the shared data space in the manufacturing work-cells, in order to virtualize devices and work-piece in a work-cell.
- The inter-process communication should be done in the form of the message passing, in order to virtualize manufacturing control processes.

Under the paradigms shown above, the infrastructural system should be designed to be the minimum system to bind program modules flexibly, and should fulfill following requirements:

- Data sharing among independent modules should be realized.
- Each module can be separately developed and compiled.
- Modules are bound at run-time and are easy to add, delete and substitute.
- Data and modules can be accessed in fully abstracted ways.
- Keeping shared data being consistent and implementing mutual execution operations.
- Models all objects in the work-cell within shared data space.
- Models all event occurrence by message sending.

2.3 Design goal

The design goals of the Glue Logic are as followings:

- The Glue Logic should assist manufacturing control software systems to have an architecture which ease integrating and coordinating multiple application program modules.
- With the use of the Glue Logic, re-using the manufacturing control system software should be easy and the software life cycle should be extended.
- The global control structure and the local control structures of the manufacturing control software system should be separated.
- In order to integrate modules flexibly, the global control structure should be executed in a interpretive way at execution time.
In order to fulfill the goals shown above, following approaches are taken:

- Implementing the manufacturing control software system as the collection of the *agents*, and realize their coordination by means of message passing.
- Use of the *active database* scheme as the shared data space among the application processes, which is a *blackboard* system with the notification message sending feature to synchronize processes.
- Making all messages to send by way of the Glue Logic, in order to make application processes independent from others.
- Centralize shared data within the Glue Logic, in order to keep shared data consistent.
- Virtualizing devices and shared data by the names, and occurrences of events by message sending.
- Virtualizing application processes by introducing the names representing them, and by implementing the feature to send notification messages on assignment to such names.

With this scheme, following advantages are expected:

- The application modules become highly re-usable and have long life cycle.
- The application modules can be written as event-driven system.
- As the relations among application modules are expressed as the message sending rules, such relations are easily re-configured and evaluated interpretively at the run-time.
- All the machine tools and the objects are represented as agents, and are represented as a name in the shared data space in the Glue Logic.
- The implementation description of the agents are hidden, and their interface become simple.
- It is easy to virtualize things and program modules in the manufacturing work-cell.

3 THE GLUE LOGIC

3.1 Architecture

The Glue Logic has been developed to support application programs by means of data sharing, event notification and condition monitoring. As the system uses inter-process communication internally over the network, the Glue Logic can play the roles of the infrastructure of the distributed manufacturing work-cell control systems. This makes development and maintenance of the event driven application easier.

Furthermore, the Glue Logic is effective not only in the distributed work-cell environment, but also in the single work-cell system, to keep application programs simple and highly independent from other program modules.

The Glue Logic is used in a configuration shown in Figure 1. In this figure, the shaded part shows the Glue Logic, and the boxes in the right half represent application processes which utilize the function of the Glue Logic.

The Glue Logic relays all inter-process communication among its application processes, and manages all data shared by those application process. Because of this, the Glue Logic
can send the change notification messages, when the values of the shared data are altered. As the virtualizing the counterpart of the communication can be achieved by relaying all of the inter-process communication, each application module can be independent from adding, deleting and altering other modules.

The application module programs can be written, using the Glue Logic API (Application Programming Interface) and a general purpose programming language, or using a global Factory Automation Programming Language (FAPL) described later. Developing new programs using FAPL is easy, but to save the existing software assets, users can convert old programs using the Glue Logic API.

3.2 Overall Implementation

In the first phase, the design of the Glue Logic is based on the client-server model of transaction processing, as shown in Figure 2, though there is no need for the users to know about its implementation.

In the prototype phase implementation, The server process of the Glue Logic is a specific process running on a specific processor. All application processes communicate only with this specific process, and there is no redundancy in this phase.

As shown in Figure 2, the Glue Logic consists of two major parts: the communication
interface subsystem and the data management subsystem. The communication interface exchanges information with other processes running in both the same work-cell controller and remote work-cell controllers connected with the network system.

The data management subsystem consists of also two parts: the data change monitor subsystem and the data management subsystem. The data storage subsystem manages the association pair of the name and the value of the object. The data change monitor subsystem monitors the changes in the data storage subsystem and sends out the data change notification messages, and executes depending data evaluation.

3.3 Behavior of the Glue Logic

The atomic element of the Glue Logic is the tuple of a name and its value, as shown in Figure 3.

The name resembles variable identifier in programming languages, and can have a value. The name is a sequence of some identifiers, separated by a period, such as abc.ijk.xyz. Using this format, users can denote data structure by the sequence of identifiers.

Using names, the application programmers can implement arbitrary data structures. In the elements of one structure, their names contain same identifier sequence in its leading part. The trailing part of their name differs from each other. The leading common part is called a stem and the trailing part is called a variant.

Each name may have some attributes. The attributes denote optional characteristics of corresponding names, and the Glue Logic changes its behavior according to the values of attributes.

As the value of the name, application programs may specify one of followings; integer, floating point real, character string, expression and link. As the name itself is not typed, users may bind any types of data in turns. If a client accesses the name bounded to an expression value, the expression is evaluated and the result is used. Using the link type value, users can point another name.

3.4 Application Program Interface of the Glue Logic

The types of the API

The Glue Logic APIs can be classified into three types. They are;

- the APIs which establish, disconnect or control communication channel, or exchange messages;
- the APIs which exchange data with the data management subsystem, or operate on the DataCell type data;
- the APIs which control the behavior of the data management subsystem.
These APIs are designed not to depend on the implementations of hardwares, operating systems, inter-process communications, or network systems.

**Network Control APIs / Communication APIs**

The network control APIs open or close the communication channels. The communication APIs wait for the message arrival, or the arrival of the messages which have some special form.

Using these APIs, the application programmers can implement all basic communications with other application programs, without having any knowledge on the inter-process communications or the network programming.

**Data Operation APIs**

Given the names and / or the attributes, the data operation APIs refer or change the value of name specified. Some of these APIs can handle multiple names and data simultaneously, and others can change the shared data safely, in order to support multi-programming environment.

Some of these APIs are prepared to handle data with the *DataCell* structure, in order to create, destruct, copy them, or manipulate fields of them. Using these APIs, the application programmers can implement a programs which do not depend on the inside of the *DataCell* structure.

**Behavior Control APIs**

These APIs direct the behavior of the Glue Logic, and implements many miscellaneous features. These include:

- The APIs which register / deregister destinations to be informed for each names. Also controls the chance to send out the information messages.
- The APIs which copy values of names to others. Some of these can copy a substructure of the name space to another.
- The APIs which inquire the names or the attributes within the Glue Logic.
- The APIs inquire the statistic information of the Glue Logic itself.
- The APIs which control the Glue Logic itself, such as doing backup / restore the contents of the Glue Logic.

### 3.5 How to use the Glue Logic

**Data Sharing**

The most simple usage of the Glue Logic is data sharing. Originally, the Glue Logic was designed as the data management subsystem of the FA programming language system. As it is very important to share some data within multiple application processes, the Glue Logic is prepared as a separate process, in order to provide common data for those related processes.

As there are some data which is strongly related, those data values should be updated simultaneously to keep those values consistent. In the Glue Logic, there are many application program interfaces (APIs) used to access or to change values of multiple names
with only one transaction. For the names which values are updated by multiple clients, there are other APIs to implement a semaphore, and to realize mutual access control.

**Data Change Notification**

In order to eliminate the needs of data polling and to decrease the network load, the feature of data change notification is used. The clients, which want to receive a change notification of a certain name’s value, can register the name of the client itself to the interesting name. The name list of the notification destination processes is kept as the value of InformTo attribute. As the clients may register other client’s name for the notification destination, the user can implement a kind of dispatcher which dispatches some processes to the events of the data change.

On the time when the Glue Logic server system receives data update request, the system searches for the clients registered as the notification destination, and then notify the fact of change to all the registered clients. In this way, the application programs are freed from polling in order to find status change.

**Automatic Update of Dependent Data & Condition Monitoring**

Some clients may need to know the value of name being a certain constant value, or the values of names satisfy a certain condition. The Glue Logic can be set to send a notification message only if a certain condition is met.

As shown in Figure 4, each name of the Glue Logic can have a dependence list as the values of Triggers and TriggeredBy attributes. If one or more elements of the list in the same name’s TriggeredBy attribute is updated, the value of the name itself is updated to have the result of an expression, which is also registered as the value of IfTriggered attribute. If this new value differs from the former value, the data change notification is sent to its notification destinations.

With this mechanism, user can implement multi-way branching by comparing value of a name to some given constants. Usually, only one of them holds true, and the corresponding application program is notified. If multi-way branching is implemented as described above, users can add other alternatives later without changing any existing application program, but only adding new condition expressions comparing a value of name to given constants. This flexibility is implemented by the Glue Logic and the registered conditional expressions.

![Figure 4](image-url)
Message Routing System

The Glue Logic can be used as a message routing system. The message to be sent is once assigned to the name in the Glue Logic by the message sender application. Then the arrival of the message is notified to the message receiver application, which should be registered as a notification destination for the name. Lastly the notified application fetches the message from the name, and interprets the message to know what is requested.

In this case, each name in the Glue Logic represents the message receiver application, and the data assigned to the name specifies the action to be taken by the receiver application. So, from the view point of the message receiver application, the name in the Glue Logic looks like a mailbox. As the message sender need not know the actual message receiver, the interface among those modules becomes simple, and the application program modules themselves become highly reusable.

In the case of selecting only one application module from many depending on the message received, those modules shares the unique name and uses condition monitoring feature to implement multi-way branching. This implementation realizes the concept of the method selection of the object oriented language system.

Furthermore, as the actual receiver application module is selected at run-time, users can change the message processor dynamically. In this way, as the message sent can be processed by the most appropriate application module for the environment at the execution time, the message routing capability of the Glue Logic makes application systems more adaptable and autonomous.

Using Automatic Process Invocation

In the case of the Glue Logic having the process invocation capability, and when the destination process of a notification is not running, the system first invokes an application program and then sends a message to the process just invoked.

This capability is not mandatory because it is usual for real-time application systems that all application programs are started at system initialization, and all are waiting for the resume signals. But in some systems which consists of many application programs, or those which can not predict the number or kind of processes precisely, it is impossible to start all programs at initialization time. And in some applications, the system may not be requested to be operated at a fast pace. In those cases, this capability is vital or useful.

3.6 The Paradigm on using the Glue Logic

In order to utilize the Glue Logic efficiently, and make application systems easy to maintenance and modify, it is important to set up the paradigm on the programming with the Glue Logic.

Execution state of processes

As the Glue Logic uses status information on the application program processes, the information should be kept and updated correctly in the Glue Logic as the value of status indicator object names.

The status indicator in the Glue Logic should have a one to one correspondence to the processes of the application program module, not to the module itself.

For each application program process, there are some states that it may take in turn.
Its status indicator represents its execution state by having predefined flag values for each state. The states are as follows, and the transitions are summarized as Figure 5:

**Initializing State** In order to start up the operation of an application program, there should be some preparation work such as initialization of the Glue Logic object names. This kind of work is done during the initializing state.

**Idling State** In this state, the application is idling, waiting for a start up message from the Glue Logic, which tells the meeting of the starting condition of the application process itself.

**Initiating State** After receiving the start up message, the application process enters the initiating state, in order to secure all shared resources it requires.

**Running State** In this state, the application process controls machine tools in the workcell, processes many kinds of data, and sends out the work-piece to another work-cell. After all tasks the application should execute are completed, the process enters the completing state. Use of this transition allows the Glue Logic to start other applications.

**Completing State** After the task completed successfully, the resources secured by the application process should be released. The completing state is used for such operation. In case of abnormal completion, the fixing procedures vary according to its internal and external status. In this case the application should export its precise status to the Glue Logic or some other fix up processes, and then exit for the next execution.

**Terminating State** If the application process receives directions to halt, it enters the terminating state. Any other application then knows that the application process has already stopped.

As described above, using status indicator object names in the Glue Logic, the application program modules can be chained or executed concurrently. This feature enables an implementation of the Sequential Function Chart (SFC) (IEC International Standard 1131-3 1993) with the condition monitoring feature (Takata, et.al. 1990) (Takata 1993). An example of the SFC is shown in Figure 6.

**Input / Output**
The read-outs of various sensors, which are required to test starting conditions for each application processes, are nice to be kept within the Glue Logic. Preparing such names in the Glue Logic, not only the application processes can use any kind of sensor read-outs
by referring to their status indicator object names, but also the starting condition of the application processes are automatically checked by the Glue Logic.

In order to reflect the status of the input in the work-cell to the status indicators in the Glue Logic, some system support processes should be especially prepared. Such process accepts interrupts from the input signal ports or scans them periodically, and updates the sensor status indicators. In some cases, some statistical operations are required in these processes. Taking a moving average value to obtain noise free data, by keeping some recent sampled values, is some of the most general operations.

On the other hand, other system support processes should be prepared in order to control the output signal by updating the values of output control names in the Glue Logic. Those processes use the data change notification feature for the output control names, and should guarantee that the assignment to such output control names are reflected to the status to the output in certain short time period. These support processes can do much more than data passing from the output control names to the output ports. For example, a robot control process can accept a motion command in the world coordinate system, and convert it to a local coordinate system if required. This is the most simple way to implement the Abstracted Manufacturing Devices.

With these system support processes, the application can access devices attached to the work-cell through the Glue Logic, instead of touching the devices directly.

4 FACTORY AUTOMATION PROGRAMMING LANGUAGE

As the language described below has not been named yet, it is referred to as “FAPL” in this paper for convenience.

The FAPL language processor is a kind of object oriented language interpreter system (Goldberg and Robson 1983), which is very powerful to model and simulate automated manufacturing systems (Bodner et.al. 1993). The language specification of FAPL is designed assuming the existence of the Glue Logic, and the language processor relies on the Glue Logic for the global data management, as well as inter-interpreter communication, process synchronization, mutual execution and condition monitoring. In other words, the FAPL language provides an application programmer view of the Glue Logic.

As each process in the application programs is executed by respective language interpreter processes, the language processor system itself has no ability to implement concurrent processing. As the process of the FAPL language interpreter can be started from the Glue Logic, the action of a certain condition can be composed in this programming language. In this case, if the condition is met, the Glue Logic server invokes the language interpreter process and sends a message to the interpreter process just created.

5 CONCLUSION

In this paper, the target and the implementation of the Glue Logic is described. The authors believe in the effectiveness of the concept of infrastructural system for agent based processing, which binds multiple tasks to be executed in the manufacturing work-cells, defined as the methods and the processes.
There are many application program systems and operating systems which have many useful tools. But the successful systems among them have sophisticated mechanisms to integrate ready-made tools to obtain fully customized tools.

The authors would like to emphasise that the smart mechanism of the Glue Logic is the very thing to make the programming system powerful and easy to be programmed, especially in the execution environment which deals with and coordinates large and complicated application programs.

REFERENCES


BIOGRAPHIES

**Masayuki Takata**, born in 1959, is an assistant professor at the University of Electro-Communications, Tokyo, Japan. He is currently studying on infrastructure systems for manufacturing control systems. His research interest includes the real-time manufacturing work-cell controls and the application of the problem solving systems to the real-time control. He received his Dr. Eng. from the University of Tokyo in 1995, on the study of the application of knowledge-base reasoning systems to the real-time control tasks.

**Eiji Arai**: born in 1953.1.15, graduated 1975 Univ. of Tokyo, Dept. of Precision Eng., doctoral course: 1979 Graduate School, Univ. of Tokyo, Dr., Eng. In 1979, Research Associate, Kobe Univ.; In 1983, Associate Prof., Shizuoka Univ.; In 1991, Associate Prof., Tokyo Metropolitan Univ.; In 1995, Prof., Osaka Univ.

Main Research Area includes: (1) Intelligent CAD/CAM Systems for Mechanical Products; Conceptual Design Support, Intention Modelling, Kinematic Simulation, Assembly Planning. (2) Intelligent and Distributed Production System Architecture; Production Process Description, Dynamic Scheduling, Active Database System. (3) Advanced Production Process; Intelligent Welding Machine, Modeling of Deformable Objects.
A framework for connecting work and information infrastructure

J. Goossenaerts
Eindhoven University of Technology, Information & Technology
P.O.Box 513, Paviljoen, 5600 MB Eindhoven, The Netherlands
email: jgo@tm.tue.nl

Abstract
An epistemological framework is defined which unifies information and command requirements of autonomous agents as they involve in enterprises and the life spans of artifacts. The framework assumes a division of the manufacturing industry domain into two sub-domains and four activity-layers. The sub-domains are the cybernetic domain and the physical domain. The activity layers cover observations, operations, improvements and innovations. The linguistic primitives records, proxies, versions and modules and the generic services of browser&report generator, model execution engine, version manager and innovation coach are assigned to the respective activity layers. Faithfulness conditions are defined between (successive) situations and events in the physical domain and corresponding constructions and transformations in the cybernetic domain. Mechanisms and principles for connecting work and information infrastructure are explained.

Keywords
Information infrastructure, CALS, framework for enterprise integration, artifact life phase service body, life cycle modelling.

1 INTRODUCTION

As computers worldwide get connected in a global network and industries move towards an extended enterprise mode of production and development (Browne et al., 1995) and learn to cope with the full life cycle of goods and artifacts (Krause and Kind, 1996), it is expected that information infrastructure services will play an increasing role for the exchange of technical and business data, and for the distributed development and control of business processes.
One can wonder how future information infrastructures for industries will look like? What services will these systems offer to enterprises, consumers and public bodies? And how will these services be realized? The MiViPoRo framework (Modules for innovations, Versions for improvements, Proxies for operations, Records for observations) presented in this paper intends to offer guidance in answering these questions. It results from a technology-independent reflection on how to offer information services for citizens and companies as they involve in artifact lives and business processes. A fundamental issue here is that the micro situation in which consumers, companies, or public bodies "work" with goods and artifacts must be linked to information, captured knowledge and computing power that is becoming globally available. An understanding of the mechanisms and principles on which such a connection can be established is necessary for creating and exploiting an information infrastructure.

The Problem Domain
In manufacturing industry an increasing number of players is becoming involved in the design, manufacture, distribution, servicing, collection at end of life, disassembly, recovery or recycling and ultimately safe disposal of artifacts and goods. These players, including companies, consumers and public bodies, have their own values and working modes. They seek to effectively meet a mix of private and public, environmental, socio-economic and sustainability challenges. Their cooperation and fair competition can be enabled by information infrastructure services for sharing information and common methods and telematics applications for order passing, contract negotiation, and cooperation in life cycle oriented product and service development.

Usage of the Framework
The MiViPoRo framework serves a double purpose. On the one hand it guides the requirements definition and development of the generic system services for an information infrastructure for manufacturing industry. The generic services are innovation coaches, version managers, secure model execution engines, and browsers and report generators. They correspond to the activity layers for innovations, improvements and operations which have been consolidated in production management practice (Inagaki, 1993). In addition the framework offers guidance for planning the future development of artifact life span oriented applications for manufacturing industries. It shows opportunities for sharing applications and information.

Related and Future Work
(a) The infrastructure concept meets requirements of the CALS (Continuous Acquisition Life-cycle Support) strategy for sharing integrated digital product data. An Integrated Data Environment (IDE) has been proposed as an end-state of the Electronic Commerce/CALS and Electronic Data Interchange (EC/CALS & EDI) vision (OSD-CALS, 1996). IDE services can be defined in the MiViPoRo framework.

(b) Regarding the capture of user requirements, the MiViPoRo framework draws on the ENV 40 003 framework for enterprise modelling (1990). The relation between the cybernetic and the physical domain has also been denoted by the term interflow as defined in (ENV 12 204, 1995). The construction of a generic plant interflow model (Goossenaerts
& Bjørner, 1994) focuses on operations inside a single production system as they pertain to a single phase – production – in the life of an artifact. The MiViPoRo framework provides a basis for expanding that construction to cater also for networks of production systems, consumers and public bodies, and for services spanning the possible lives of artifacts.

(c) The MiViPoRo framework treats only the generic aspects of a framework for connecting work and information infrastructure. In order to cater for manufacturing in all its variety and particularity, the framework should be extended and applied. Artifact (life phase) models should be extended by drawing on progress in product modelling (Krause et al., 1993) and the related STEP standards (Gielingh, 1993). Process models should be enriched by drawing on CIMOSA (AMICE, 1993).

(d) A detailed study of the transformations of possible lives models, and the manner in which an innovation coach (preferably supporting a concurrent engineering mode of design) could support it, could be based on the General Design Theory (Yoshikawa, 1981), as indicated elsewhere (Goossenaerts et al. (1996)). Innovation and improvement layer transformations and the realization of new "interflow" systems in enterprises – business reengineering – on the basis of new or revised modules are part of enterprise life cycles (Williams, 1994).

(e) For its operation layer, the framework assumes mechanisms for the computer network based execution of enterprise and artifact possible lives models. The concepts of interflow system and the execution mechanism for enterprise formulae (Goossenaerts, 1993) offer some relevant insights. Other inputs are expected from advances in system interoperability based on object-oriented interfaces and the developments on open distributed processing (ISO/IEC DIS 10746-3).

2 THE MiViPoRo FRAMEWORK

An information infrastructure system for manufacturing industry is a large and complex system without centralized control. It supports and enables industrial processes that are purposeful (answering the why question), proceed in time (when), are situated in chains of cells (where), depend on the initiative and effort of agents (who), and transform or use artifacts and materials (what). These processes show forms of cooperation between several players having their own independent interests, values and modes of operation with respect to artifacts, goods and services. The MiViPoRo framework supports straightforward and infrastructure-wide reasoning and computation on transformations, queries and answers involving purpose, (points and intervals in) time, places, agents and artifacts and their attributes, as knowns and unknowns.

The present description of the framework looks at the linguistic primitives that are typical for the different activity layers, and the faithfulness conditions on the physical and cybernetic domain that are defined using them. The meta-linguistic connections between different primitives are not covered in detail.

2.1 MiViPoRo, its Generic Services & Linguistic Primitives

The MiViPoRo framework divides the problem domain of manufacturing industry into
Part Three  Integration Frameworks and Architectures

two sub-domains and four activity layers (Figure 1). The sub-domains are: the physical domain comprising the physical space, time and matter, with artifacts, goods, agents and cells (spatial units) having their lives in it; and the cybernetic domain which adds memory, communication, monitoring (including knowledge-capture) and control (computations) services to the physical domain. The term interflow (short for interactive flow) denotes the coordination and monitoring of time&space&matter situated physical processes by means of computational processes. The concept of physical process is wide and includes processes which are (usually) not modelled (such as for instance the thought processes of individuals who work in improvement or innovation layer). The concept of a computational process is kept narrow, it covers only transformation or storage of digital representations as they are processed by computer programs.

The four activity layers span the two sub-domains and cover observations, operations, improvements and innovations. In each of these layers, work – action in the physical domain – has to be connected to computations and communications in the cybernetic domain. This is done by the generic services: innovation coach, version manager, model execution engine and browser&report-generator. These services provide interfaces between objects existing on the physical and cybernetic sides of the activity layers.

<table>
<thead>
<tr>
<th>Physical Domain</th>
<th>Generic Services</th>
<th>Cybernetic Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Innovation Layer</td>
<td>Innovation coach</td>
<td>Module version manager</td>
</tr>
<tr>
<td>Improvement Layer</td>
<td>Version manager</td>
<td>Module version of module</td>
</tr>
<tr>
<td>Operation Layer</td>
<td>Model execution engine</td>
<td>Proxy (mobile occurrences of possible lives model)</td>
</tr>
<tr>
<td>Observation Layer</td>
<td>Browser &amp; report generator</td>
<td>Records (on proxies w.r.t. modules or possible lives models)</td>
</tr>
</tbody>
</table>

**Figure 1.** Activity layers, generic services and linguistic primitives in MiViPoRo.

On the physical side, the term entity denotes an artifact, agent (person, machine), or place which has a definite physical existence in itself. At any point in time, an entity is present at exactly one place. It is characterized by an existence in matter and a mobility in space (except for some places relative to each other). An agent has powers which determine the actions it can execute. Entities are grouped in classes on the basis of similarities in their time&space&matter situated possible lives. At a point in time, the place where an entity is, will accommodate certain possible or required life phases of the entity. This set of life phases is called the local type of the entity and determines the next events in which the
entity may involve. An entity has a life span, a succession of space-time situated events in which other entities may be involved. The (local) type of an entity will change during its life time.

The linguistic primitives on the cybernetic side are: (i) modules are created and transformed in the innovation layer. One module typically captures time-space-matter independent aspects (knowledge) on one possible or required phase of the life span of a class of entities. Modules are combined into a possible lives model of the class. (ii) Versions of modules and possible lives models capture modifications as they are performed in the improvement layer. (iii) A proxy exists in the cybernetic domain as a unique representant of a physical entity. It captures knowledge on the entity and the modules (each belonging to the possible lives model of the class of the entity) for which it has been earmarked. Proxies exist at the operation layer. (iv) A record on a proxy is created in memory of the proxy’s past or current flow through the cybernetic domain, in reflection of the time-space-matter situated life span of the corresponding entity. Records are recorded during operation layer activities, for further use in observations.

The activity layers, generic services and primitives provide us with a basis to specify common arrangements concerning artifact lives and the interaction between processes in the physical domain and processes in the cybernetic domain.

Simplifying a bit, one could state that in the physical domain, each player has a territory where its entities exist, or where its modules prevail. An example of the first case is the citizen who keeps belongings in a house or locker. The latter case is illustrated by a public authority who issues the rule that security belts should be present on the seats in motorcars, and defines conformance tests for this rule. The modules managed at the public body constrain the outlook and usage of cars. Similarly, each player owns a fraction of the MiViPoRo-conformant "cyberspace". These are the own proxy and the proxies of the entities that are owned by or entrusted to him or her. The cybernetic domain of the manufacturing company includes modules of the production processes it can sustain. It also includes proxies of the machines, products in stock and in production, and performs the production planning and control functions. The cybernetic fraction of a public body could include modules specifying rules and procedures to enforce them, as well as files and registers listing records on the subjected entities (and proxies) in a territory.

2.2 Mi-:Modules for Innovation

Innovation processes use modules for the time&space&matter-independent modelling of cells, artifacts and agents. Modules are the first deliverables of innovation processes. A module describes a possible or necessary phase in the possible lives of one or more artifacts, agents, and/or cells. In an industrial context where the use of product and process models is mature (e.g., one uses digital mock-ups of products and process models) and material and work are expensive, innovation processes will create and evaluate modules first, before physical entities are produced, or processes are carried out in reference to the modules. An innovation coach is a generic service which supports design, for instance in a concurrent engineering style, with global sourcing of design data (modules) and artifact data (proxies and records).

Definitions: (i) A module is a space&time&matter independent description of a possible or required phase in the life span of one or more entities. It describes what one
should know to handle an entity in a certain situation. If a place has been equipped to accommodate a life phase of an artifact (e.g., an assembly line is equipped to produce a type of car) then the place is said to be earmarked for the life phase. On the cybernetic side the place’s proxy is earmarked for the module describing the life phase. If an entity enters the place (e.g., a car enters the repair shop) earmarked for one of its possible life phases, then the entity’s proxy will be stored at the cell proxy, and the module’s definition will determine the activities for the entity.

(ii) A possible lives model is a set of modules which have been defined for a class of entities, or in which the entities of the class can be involved (when a module describes a phase in which entities of different classes are in a relationship). Modules may describe concurrent, successive, alternative or mutually exclusive phases in the possible lives of entities. For instance, the possible lives model of a product could include modules for the phases: need evaluation (prefeasibility, feasibility, project definition), design & development, production, transportation, usage, maintenance (preparation, repair, modification, supply), and dis-engagement.

(iii.a) The capability of a cell is a set of life phases which are accommodated at the cell. It is described by modules in the cell’s proxy. In principle there is no limit on the number of modules in a cell proxy (a garage may be equipped to repair cars of several brands and types).

(iii.b) The type of an entity is the time & space dependent set of the artifact life phases in which the entity is participating. It is described by all modules for which the entity is earmarked. In principle there is no limit on the number of modules in the type of an entity. The type of an entity may change as time proceeds and the entity moves through the physical domain. E.g. a person who is citizen of one country, becomes a visitor of another country when crossing its border. While being entitled to engage in a labour contract in the own country, the person may be excluded from this right in the other.

(iv) The term typogeny (combining type and -geny) denotes the origin, development and evolution of a type.

(v) The term typology denotes the theory or study of types and typogenies. It addresses the relations between typogenies, and their module-wise changes as new versions of artifact types, agent types and cell types are developed.

Innovation depends on the transformation and appraisal (to select among alternatives) of the modules which are part of possible lives models. New modules must be better or enable new functions while respecting boundary constraints (technology, economy, regulations, mechanisms of nature).

Innovation coach systems can support innovation by allowing one to unify modules (from registers, own or subcontractor warehouses or servers) which are relevant for innovation in a certain life phase (production, maintenance, use, disposal). The design engineer can then replace, refine or improve the module. The coach should offer specific functions to evaluate proposals w.r.t. objective criteria, such as conditions for production, transportation, distribution and recycling.

A possible formalism to express modules is that of enterprise formulae as proposed by the author (1993). Enterprise formulae feature high information density, modular structure and strong context integration. This formalism allows one to combine in a
single modular run-time independent expression, the three views that are commonly used to reduce the complexity of enterprise process models, i.e. the data view, the function view and the organization view (see Scheer, 1994, pp 10-13).

2.3 -Vi-: Versions for Improvement
Improvement processes develop new versions of existing modules, proxies or possible lives models. A version manager supports modifications while guarding compatibility among the modules and the proxies created in reference to them.

2.4 -Po-: Proxies for Operation
Operations comprise agent-artifact-infrastructure activities which are situated in the physical and cybernetic time&space domain and which create and transform entities and corresponding proxies (and records). Entities are occurrences – exemplars or instances – of the classes, their proxies progress within the constraints set by a possible lives models. Activities may require the retrieval, glueing together and transformation of information (on proxies, modules, records) in the cybernetic domain, or may be driven by it. Typically, the retrieved information will underpin a decision. Transformations will update data kept in accordance with constraints and possible behaviours defined in modules.

Definitions:

(vi) A proxy or information proxy is a time-space situated linguistic object which exists in the cybernetic domain and is given the exclusive role to represent a unique current or future time-space-matter situated entity of the physical domain. An artifact proxy represents an artifact, an agent proxy represents an agent or (legal) person and a cell proxy represents a cell (or spatial unit). For each entity there is at most one proxy, by definition. Proxies and the corresponding entities exist independently from the life phases (modules) for which they can be earmarked, and unmarked (the type of the proxies can change). They can have a local relevance (e.g., in a production system, a part’s proxy may be removed after it has been assembled in an assembly). A proxy may hold a reference to the modules for which it has been earmarked. A cell proxy that holds a module will have records on the proxies that are earmarked for the module.

(vii) The term ontogeny (combining onto- (ens) and -geny) denotes the origin, development and evolution of a single entity. Ontogeny denotes the progression of an entity as it is earmarked for certain life phases and unmarked. At any point in time and place, an entity may be committed for a number of life phases. This is reflected in the cybernetic domain by the proxy being earmarked for the corresponding modules.

(viii) The term ontology denotes the theory or study of ontogenies, i.e. of those properties – having an origin, development and evolution – that all entities have in common.

In the context of a manufacturing system, the materialization process (production) will map a proxy of a future entity to the entity; this process is guided by the (production) modules in the possible lives model, and executed at cells applying their capabilities. A proxy of a future entity may be created in response to an order, the production process is executed in accordance with the explosion of the order as described by Goossenaerts and Bjørner, 1994.

Possibilities for the shared use of modules are illustrated: The possible lives model car
includes the models $X_{car}$ (car of brand X), as well as $Y_{car}$ (car of brand Y). $X_{car}$ and $Y_{car}$ differ from each other because they are produced according to different production modules and have to be used according to (partly) different usage modules. Car entities that are earmarked for market A should meet particular regulations (safety, environmental, performance) and rules for trading and using them. This suggests the shared use of the corresponding module (e.g., module: $A\_market\_rules\_for\_cars$) for both the $X_{car}$ entities and $Y_{car}$ entities for market A. If a car entity is earmarked for the market A it will be registered in reference to the module $A\_market\_rules\_for\_cars$. Entities of the class car can be created in reference to a possible lives model composed from a number of alternative modules (such as $X_{car}$, $A\_market\_rules\_for\_cars$, etc.). Changes affecting one or more life phases of car, can be localized in a small number of modules. Per module, past and current versions are to be managed. Compatibility rules must be observed for all modules defined for a class.

**Definition:** (ix) An interflow event is the constituent event of a process in the operation layer. The event may involve one or more entities, their proxies and relevant modules and records. It achieves a synchronous transition between successive situations at a physical cell and the corresponding transition of the proxies and records in its (cell) proxy. For non-trivial events, the following protocol can be implemented: the pre-conditions of the event cause an infrastructure-wide collection of relevant records and modules (life-histories and possible lives are constructed). Eventually the pre-conditions are verified, and the modalities of the transaction (e.g., the price of a purchase) are determined, then it is committed. Finally, the post-conditions of the transaction result in the sending of messages (with or without delay) informing the relevant public and private bodies of the transaction and its modalities (e.g., the tax authorities will be informed of the value added of the transaction and accordingly collect taxes, etc.).

### 2.5 -Ro: Records for Observation

A single entity has a unique physical manifestation in the physical world (at any point in time it can only be present at one place), similarly, a proxy has a unique manifestation in the information infrastructure. In contrast with the unicity of a proxy for an entity, several records denoting the same entity may exist in the information infrastructure, typically one per module or life phase for which the entity (and its proxy) is or has been earmarked.

**Definitions:** (x) A record is a unit of information (text, number, picture, sound) that is kept in reference to a module, on an entity and its proxy, after the entity has been earmarked for the module (i.e., it has been decided that the module will describe (or prescribe) a phase in the life cycle of the corresponding entity). Modules that describe relationships (e.g., the marriage relationship) will hold records on all entities and proxies that are related to each other according to the module.

(x) In the cybernetic domain, observation is the act through which an agent reads a record without affecting its value. In the physical domain, observation may involve the measurement of one or more properties of an entity, for instance with the purpose of entering data into a computational system.
2.6 Faithfulness between Physical and Cybernetic Domain

The linguistic primitives and their constructions give rise to different faithfulness relations between actual and possible situations in the physical domain and corresponding constructions in the cybernetic domain.

**Definition:** (xii.a) There is observational faithfulness between on the one hand: a current, past or future situation or a succession of situations in the physical domain ((successive) arrangements of entities); and on the other hand: a construction or succession of constructions in the cybernetic domain ((successive) arrangements of proxies and records), when, in terms of a number of modules, the records and proxies, the constructions in the cybernetic domain correctly describe the (succession of) situation(s) in the physical domain.

(xii.b) There is operational or ontogenic faithfulness between physical flow of entities in a physical domain, and computational flow (flow of proxies, change of records in a cybernetic domain), when, in terms of a number of modules, the successive transformations of the proxies in the cybernetic domain correctly describe the transformations in the physical domain.

(xii.c) There is model or typogenic faithfulness between arrangements of entities in a physical domain, and a model – consisting of modules, capabilities, types and proxies – in the cybernetic domain, when, the model correctly describes the entities, their capabilities and types, in the physical domain, and therefore enables the operationally faithful flow of physical and computational processes.

Given observational faithfulness between initial physical situation and cybernetic construction, then, operationally faithful interflow will achieve observational faithfulness at succeeding points in time. Operationally faithful interflow builds on typogenic faithfulness between models – proxies and enterprise and artifact possible lives models – and arrangements in the physical domain. It also assumes a regular interaction between the cybernetic domain and the physical domain.

2.7 Achieving Operational Faithfulness: Cyber-networks

A pre-condition for creating an information infrastructure is that one is capable of achieving operational faithfulness between activities in the physical domain and computations in a MiViPoRo conformant cybernetic domain. Because operations are distributed in space it is necessary to handle the interface between the two domains by means of a network of cell proxies, offering some of the generic interface services between the two domains. Such a cell proxy is called a cyber-cell. A network of cyber-cells is called a cyber-network.

**Definitions:** (xiii) A cyber-cell is a system that offers one or more of the generic services, and has the capability to communicate with other cyber-cells. The agent proxies at the cyber-cell represent the entities that are working at the place and the artifact proxies represent the artifacts that are there (being worked on or with). The records at the cyber-cell stand for proxies which have been or may be at the cell, or which the cyber-cell knows about. Agents and artifacts can enter and leave the cell, and likewise do proxies enter and leave the cyber-cell.

(xiv) A model execution engine is a software system that manages the computational side of operational faithful interflow at a cyber-cell. It will manage the access rights of
agents, the access possibilities of artifact proxies, and the transformations of proxies and records held in reference to the modules. Modules, together with the proxies present in a place, and stored records will determine or constrain the behaviour (possible events) of the entities and their proxies. The possible correlations of events are derived from the behaviours as defined in the modules (locally) and their grouping in possible lives models.

\((xv)\) A artifact life phase service body \((or\ ALPS)\) is a cell which undertakes to provide services – with both cybernetic or physical aspects – for a number of life phases of a number of entity classes. When it has a cyber-cell such that there is typogenic faithfulness between its modules and the capabilities of the cell, it can achieve a local connection between work during artifact and agent lives and an information infrastructure. The cyber-cell manages the local computational flow and the communications with other ALPSs. Examples are garages which offer repair services for cars, and national registers of private citizens which determine, among others, which adult citizens can participate to elections.

\((xvi)\) A cyber-network is network of cyber-cells. It interfaces a distributed construction of modules, versions, proxies and records that constitute a MiViPoRo conformant cyberspace with users in the corresponding physical domain. Model execution engines at the cyber-cell may implement protocols in support of the global sourcing of possible lives models and life span data in support of (operation layer) decisions and work, as well as the global propagation (to other cyber-cells) of the consequences of such decisions.

2.8 A Cyber-network as an Information Infrastructure

An information infrastructure is a cyber-network which is maintained by a loose federation of agents – companies, public bodies and private citizens – and which is constructed such that the services of its corresponding ALPSs extend over the possible life phases of a wide range of artifact types as well as over the relevant events in the lives of artifact occurrences \(\text{(artifact histories)}\): for each possible or required phase in the life of any artifact and for the agents involved, there are ALPSs that can provide the relevant support. The infrastructure, seen as a whole, constructs and maintains a dynamic map of past and future artifact and agent motions and transformations, as they matter for achieving individual or public goals in the territory. The infrastructure is distributed and should show robustness, modularity, maintainability, simplicity and efficiency. It should offer a coherent access to services. It should show typogenic faithfulness for a wide range of manufacturing resources and artifact types in a territory and achieve operational faithfulness for the lives of companies, consumers, public bodies and artifacts in the territory. Preferably with a minimal duplication of modules and data and cost of communication. An information infrastructure supports the global sourcing of artifact and process model data \(\text{(modules)},\) artifact histories \(\text{(occurrences)}\), and workflow elements \(\text{(occurrences)}\) in support of identification, decision, planning and action, prior to the proper propagation of the consequences of decisions and actions. It plays an enabling role for any major transaction involving private citizens, public bodies, companies, products or materials, or their models. If based on streamlined and harmonized product and artifact and enterprise models and data, the combined service offered by the information infrastructure takes away major non-productive burdens from \(\text{(small and medium) enterprises and consumers,}\) while providing them with the relevant information, on the spot, whenever they need it.
3 MiViPoRo AND ENV 40 003

Several aspects of MiViPoRo have been influenced by the framework for enterprise modelling ENV 40 003. Only the differences are highlighted.

(a) Possible lives models in MiViPoRo are intrinsically distributed and oriented towards the reuse of partial models (modules).

(b) In the dimension of Model of ENV 40 003 (with stepwise derivation from requirements model, over design model to implementation model) MiViPoRo emphasizes the requirements model. It anticipates model execution engines working directly with possible lives models, including those of ICT resources (processors, networks and peripherals).

(c) In the dimension of views, ENV 40 003 selects the views Organisation, Resource, Information and Function. This selection has been biased by traditional solution techniques. In comparison MiViPoRo's concepts are closer to the problem domain:
   - the general categories of agent, artifact and cell and the involvement of agents and artifacts in time-space-matter situated events: the category of cell reflects the organisation view of ENV 40 003 cells have capabilities; agents group the functions which they can execute, and artifacts can be present as resources or products (different modules from their possible lives model).
   - MiViPoRo emphasizes a distinction between the information that is kept on artifacts, agents and cells on the one hand, and the physical manifestation of these entities on the other hand. Thus all model-components (modules, versions, proxies and records) belong to the information view.

(d) ENV 40 003 assumes that there are generic (basic) constructs from which partial and particular constructs are derived. MiViPoRo's generic concepts can be identified in any model-based infrastructure system, or components of it. The generic concepts include: (i) the distinction between the cybernetic domain (modules, versions, proxies, transactions and records) and the physical domain (classes, entities, events, processes); (ii) the concepts of agent, artifact and cell; (iii) the concepts life phase, occurrence and possible lives model (a possible lives model is described by modules that meet certain syntactic rules of well-formedness); (iv) the modalities for the interaction of agents, artifacts and cells (events involve agents and artifacts and are situated in space, time and matter); (v) the distinction between static model describing possible dynamics, and the dynamic model (the "run time") which changes in response to physical flow, conform the static model.

(e) The role which partial models play in ENV 40 003's stepwise particularization process, is replaced in MiViPoRo by the role of cell-proxies and the modules which they contain. If one company manages modules for certain life phases of artifacts, then its suppliers or customers can define own, complementary modules for the life phases to which their core competences pertain.

(f) ENV 40 003 announces executability of enterprise models, but lacks the views that model execution engines and innovation coaches will have on the models. The MiViPoRo framework supports a more precise definition of the concept of a "model execution engine". A cyber-network plays for (distributed) possible lives models the same role as the "universal computing machine" (or Turing machine) plays for a computational algorithm.
Computations in the cyber-network are different from those of a universal computing machine because of the distributed memory, non-determinism, concurrent computing, dependability, ... 

(g) The MiViPoRo innovation process combines – in a pragmatic way – features of step-wise derivation (ENV 40 003 dimension of model) and particularization (dimension of genericity).

4 CONCLUSION AND FUTURE WORK

The MiViPoRo framework supports one’s reasoning about information infrastructure services for industry. It considers enterprise and artifact models as sets of compatible modules and offers guidelines for the definition and development of model execution engines, browsers and innovation coaches, and version managers. The definition of an information infrastructure – drawn from the framework’s concepts – offers some theoretical application and validation of the framework. Further applications and validations should be more practical and address: the use of international standards for product and process modelling in the definition of modules; the specification of model execution engine and innovation coach (considering standards such as those for electronic data interchange and open distributed processing); the precise description of cybercells (including the protocols for interactions with human and other agents, for mutual communication and computation); and demonstration projects.

5 REFERENCES


An Extensible Reference Model for Data Integration

G.N. Benadjaooud and B.T. David
Ecole Centrale de Lyon, Laboratoire GRACIMP, Dépt. MIS
B.P. 163, 69131 ECULLY CEDEX, FRANCE, Tél : (33) 72.18.64.42
Fax : (33) 78.33.16.15, email : benadjao@cc.ec-lyon.fr, david@cc.ec-lyon.fr

Abstract
The DEE (Data Exchange Environment) is a platform for the integration of isolated applications. It co-ordinates and controls the data exchanges between the different applications. The DEE uses an Extensible Reference Model (ERM) that gives a global and unifying view of the manipulated data. The MO-MER Model is an object oriented model, facilitating the extendibility of the ERM, because its Meta-level allows the addition and the modification of the manipulated concepts. The extendibility of the ERM is ensured by schema evolution methods that go with the MO-MER model. These methods allow the update of the ERM during its life cycle.

Keywords
CIM (Computer Integrated Manufacturing), integration, databases, data exchanges, data modelling, Object Oriented Model.

1 INTRODUCTION
Data-processing in manufacturing systems is used as an assistance tool for the elaboration of the products along the whole production process. Among these tools we will quote : Computer-Aided Design (CAD), Computer Aided Process Planning (CAPP), Computer-Aided Manufacturing (CAM) and Computer Aided Quality (QAO) systems. Due to the fact that these application software packages could be conceived and developed independently, they often remain disparate and isolated.

The data integration of isolated applications usually employed is to let applications share and have access to the common data. Two approaches mainly used are either based on the application interfacing and the use of standard format or based on the federated databases.

Interfacing applications is to create a data translator between two applications that have to exchange data. To ease this interfacing and to reduce the number of translators, standard formats have been defined, like IGES (Initial Graphics Exchange Specification), SET (Standard d’Echange et de Transfert) (Wix. 1986, Scholz-Reiter. 1992) and STEP (Scholz-Reiter. 1992, Brun. 1992). They harmonise the data expression between applications. To exchange data using standard formats, the data are translated from the source application format to the standard format and from the standard format to the target application format. So, for each application two translators are needed : one for the data transfer from the application to the standard format, another one for the data transfer from the standard format to the application.
2 OUR APPROACH

We fixed as an objective the study of possibilities of integration of the isolated applications. Our aims were:

- to ensure a global data perception by conceiving a data referential. Its purpose is to express semantic links that exist between the manipulated information. It serves as an informational integration support. It is a general data model able to be enriched by time. This is why we call it Extensible Reference Model (ERM).
- to allow a reliable flow of all kinds of data between applications that can be geographically distanced and to use different data managers.
- to allow also a progressive data migration from applications to the ERM.
- to preserve existent applications that are valuable acquisitions of the companies.

**Figure 1 The Data Exchange Environment**

The solution that we present is an environment for data exchanges called DEE (Data Exchange Environment). It is a platform for the integration of isolated applications that allows a coordination and a control of data exchanges between the different applications. The DEE has an Extensible Reference Model (ERM) that allows it to have a global and unifying view of the data manipulated by connected applications and also allows us to solve the modelling and the perception conflicts.

To preserve the existent, the connection of applications to the DEE is a weak coupling; it is done by using local interfaces (L.I.) (Benadjaoût 1996). These allow applications to communicate with the DEE and they allow the DEE to overcome problems of differences of DBMS (at the data model level and data manipulation language level). They have the following functions:
• the data transfer from the application to the DEE and vice versa,
• the elimination of data representation and data access differences that may exist between applications and the DEE,
• the encapsulation of the application data access by harmonising accesses to data. Thus they ensure the independence of the DEE towards the connected applications.

In this paper we will focus our presentation on the ERM and the MO-MER model. The details on the architecture of the DEE can be found in (Benadjaoud, 1996).

3 THE ERM

The ERM is a unifying schema of information processed by the different applications. It eliminates conflicts of perception that may exist in the real entity as represented by different applications.

To allow the ERM to represent complex entities, to support different viewpoints of a same entity, to express and to maintain the coherence of static and dynamics data, we have developed an object oriented model, called MO-MER that serves as a canonical representation model of data. Thus, applications manipulating complex entities, such as CAD systems and CAM systems, can be connected to the DEE as well as other future applications that will use object oriented DBMS.

The ERM is a reference for all applications connected to the DEE. It offers a global view on shared objects and relationships between them. It is also a reference for the DEE so that the latter could undertake its data distribution functions.

Adding to the global schema of shared data that we call shared schema, the ERM contains for each application connected to the DEE an export schema that represents the data that the application is ready to share with others. These export schemas define the contribution of applications to the federation. Correspondences between export schemas and the shared schema link each entity in the shared schema to its correspondent in export schema. To each correspondence are associated one or several predicates that define conditions to be satisfied to make this correspondence valid.

The design of a ERM is made in an incremental manner; we create from existent applications (and specific models) a part of the ERM that will then be enriched as a new application will be connected to the DEE.

The ERM offers a set of access functions allowing the DEE to obtain information concerning the shared schema and correspondence between entities. These functions allow, for example, to know the attributes of an object, the correspondence between two objects, the type of an attribute, etc.

4 THE ERM-INSTANCES

The ERM-Instance allows us to gather a determined set of shared data. Data contained in the ERM-Instance can represent:

• data of the new applications connected to the DEE,
• data of applications that are not well organised (the case of applications using files), and/or
• the most shared data.
In the first case, the new applications are the ones developed specifically for the company that may use the ERM-Instance as a data manager. It means that the new applications non-specific to the company, that possess their own data manager, will continue to use their data managers and will be connected to the DEE through local interfaces.

In the second case, ERM-Instance allows applications to put their data together, a semantics modelling of their data with the expression of links between data. Thus, the storage of data of the same project in a unique storage system, for example, is made possible. This will facilitate, thereafter, the re-use of data, but a re-engineering of data is necessary; create the conceptual schema from the inputs and the outputs of each application. The transfer of this data project from the application to the ERM-Instance is made, gradually, by using the DEE functionalities.

The DEE, in this case, can be seen as a tool for a progressive migration of non structured data to a central database (ERM-Instance) by undertaking transfers of data from applications to the ERM-Instance.

In the third case the ERM-Instance groups objects that are the most required by applications. These objects constitute a database that groups global characteristics of used objects. It allows:

- the limiting of the data redundancy and the data incoherence,
- the saving of time for the users in the preparation and the transfer of data.

5 THE MO-MER MODEL

The MO-MER Model is an object oriented model, facilitating the extendibility of the ERM, because its Meta-level allows the addition and the modification of the manipulated concepts. The extendibility of the ERM is ensured by schema evolution methods that go with the MO-MER model. These methods allow the update of the ERM during its life cycle.

To take into account the future DBMS that could be used by new applications to connect to the DEE, we have made ensured that the basic concepts manipulated in MO-MER converge to those described in Object Database Standard developed by ODMG (Atrood. 1993).

5.1 Presentation of MO-MER

```plaintext
Class WoodScrew

properties of the class
SuperClass: Screws;
keys: SerialNumber;
Constraint: Diameter/Length < 1;

properties of instances
SerialNumber: Integer;
Head: (Round, Flat);
Diameter: Real;
Length: Real;
Matter: String;

operations on instances
Price (): - > Real /* returns the Price of live it that is calculated in function of
the diameter, the used matter and the length) */

Figure 2 The structure of a class
```
The MO-MER object model is presented as follows:

- The basic concept of the modelling is the object. Objects are grouped in classes. All objects of a same class have the same behaviour and the same characteristics.
- The behaviour of an object is defined by a set of methods described in the class.
- The state of an object is defined by values of its characteristics. These characteristics can be attributes of the object or relationships with other objects.

The class is the conceptual entity that describes the object, it groups a set of objects (instances of the class) that share common characteristics (described by attributes) and have the same behaviour (described by operations). The example in figure 2 is a description of the class WoodScrew. The class WoodScrew represents the set of objects having a Diameter field that is an integer, a Head field, a Length field and a Matter field.

The relationship between an object obj and its class C is a "IS_A" relationship, and obj is an instance of C. For example the screw whose value is the tuple (1234, Round, 2.5, 10, steel) is an instance of the class WoodScrew.

From a class, it is possible to define other classes (sub-classes) more specific (specialised) completing characteristics of their mother class using the inheritance mechanism offered by the MO-MER model. The class WoodScrew, for example, as defined previously, is a sub-class of the class Screw. It inherits the attributes Diameter and Length.

A specialisation can be multiple due to the multiple inheritance that allows a class to have several direct super-classes, and to inherit the union of attributes and methods of its super-classes.

The MO-MER offers constructors such as List, Bag and Set that allow an attribute not only to have mono-valued values but also multi-valued values.
- The Bag is a collection of objects that authorises the duplication of objects.
- The Set is a collection of objects that does not authorise the duplication of objects.
- The List is a collection of objects that authorises the duplication of objects and institutes an order relationship between objects.

Two classes are related if the domain values of an attribute of one class is the set of objects of the other class. For example, to express the relationship MilledBy that connects a WoodScrew to its milling machine, we add to the class WoodScrew an attribute called MilledBy whose domain values is the class milling machine. To express the same relationship in the opposite way, we add to the class Milling Machine the attribute Mills whose domain values is the class WoodScrew. The constructor Set, in this case, has allowed us to express that n WoodScrews are connected to a Milling Machine. Thus relationships [1,1], [1, n] and [n, n] can be expressed by the constructors Bag, Set and List.

Two objects are related if one of them is a value of an attribute of the other. For example, to express the relationship MilledBy that connects a wood screw to a milling machine, we set the attribute MilledBy by the identifier of the object milling machine, by which the screw has been manufactured.

(obj4, (SerialNumber: 021425, Head: Round, Diameter: 03, Length: 8.5, MilledBy: obj8))
(obj8, (ToolNumber: 14, MillingTime: 14.03)).

The MO-MER distinguishes two types of relationship: exclusive relationships and non exclusive relationships.
Class WoodScrew

properties of the class
SuperClass: Screws;
key: SerialNumber;
constraint: Diameter / Length < 1;

properties of instances
SerialNumber: Integer;
Head: (Round, Flat);
Diameter: Real;
MilledBy: MillingMachine;
Length: Real;
Matter: String;

operations on instances
Price (): - > Real;

Class MillingMachine

properties of the class
SuperClass: Tool;
key: ToolNumber;

properties of instances
ToolNumber: Integer;
Mills: Set <WoodScrew>;
MillingTime: Real;

operations on instances
NumberScrewByDay (): - > Integer;

Figure 3 A relationship between two classes

Exclusive relationships: a relationship between two objects is exclusive if the two objects cannot be connected by the same relationship to other objects. This type of relationship allows us to model the relationship composed-component that connects a complex object to its components.

Non exclusive relationships: a relationship between two objects is non-exclusive if the two objects can be connected by the same relationship to other objects.

To illustrate these two types of relationship, let us take the following three classes:

Class Car ....
- EquippedBy: Exclusive engine;
- ManufacturingSupervisor: Person;

Class Engine

Class Person

The relation EquippedBy that connects a car to the engine with which its equipped, is an exclusive relationship, because an instance of the class Engine MO can be connected only to one instance Car V0. An other instance Car V1 can not have the instance MO as a value of EquippedBy.

The relationship ManufacturingSupervisor that connects a car to the person who has supervised its manufacture, is a non-exclusive relationship, because a same person, Jean Claude, can supervise the Car V0 and the Car V1.

A conceptual schema in the MO-MER Model is expressed by:
- the specification of the set of classes that compose it,
- a lattice of specialisation whose arcs correspond to the relation super-class / sub-class,
- a graph of relationship whose arcs correspond to relation reference to.

The example illustrated by the figure 4 represents the specialisation lattice of the class screwers whose sub-classes are the class Screw and the class Nail. The root of this lattice is the class Object.

Relationships [1, 1] between two objects are represented by a simple arrow and relationships [1, N] are represented by double arrows. Relationships [N, M] are represented by double arrows on each extremity of the line. The example illustrated by the figure 5 is a graphical representation of the example illustrated by the figure 3 where a milling machine is in relation with several Screws.
5.2 The meta-circularity of MO-MER

The model is based on a uniform architecture of three levels: instance/class/meta-class, where all is object and the meta-classes are real classes in the sense that they can have instances and sub-classes. This architecture facilitates, on the one hand, the evolution of the ERM which is important for the openness of the DEE to new applications, and on the other hand the extendibility of the basic classes used in the ERM in the case where new concepts (new semantic relationships between objects, new types, etc.) are used by new applications.

The MO-MER is a reflexive model. It allows the ERM to manage information concerning itself, and this offers several advantages. The first one is what we call the uniformity of representation; the primitives of the model are used to manage all information including the meta-information. The second is access and manipulation uniformity; the extraction of data that is managed by the same access primitives of the model whatever the type of the information or its state.

5.3 Correspondence

The ERM is constituted of a shared schema representing all objects manipulated by applications and a set of export schemas. Each export schema represents manipulated and exported data of an application.

A correspondence links a class of an export schema to its equivalent in the shared schema. For a given class in the shared schema, the correspondence enumerates attributes of this class that can be found in a given export schema. It plays the role of attribute filter.

To avoid the redundancy of representation of the same class in the shared schema and in the export schemas, classes of export schemas are not represented as such. An export schema is represented by a set of correspondences. Each of these correspondences is characterised by the class of the shared schema to which it is linked (associated class), a list of attributes of this class exported by the export schema and the predicate that has to be verified to validate this correspondence.
5.4 An example of a modelling in MO-MER

Figure 6 illustrates an example of an ERM. The instruction Create ERM announces the creation of an ERM whose name is given. The following described classes constitute the shared schema of the ERM. The instruction Create export schema announces the creation of an export schema whose name is given. The set of the followed described correspondences constitute the export schema.

/* ERM for car design */
Create ERM CarDesign;
Class Car
{Class Properties
  Superclass : Object;
  Key : No_Ref;
  Constraint : No_Ref > 99999 et No_Ref <= 999999
  Instance Properties
    No_Ref : Integer;
    EngineCharacteristics : Engine;
    BodyCharacteristics : CarBody;
    OptionCharacteristics : Option; }

Class Engine
{ Class Properties
  Superclass : Object;
  Key : No_RefEngine
  Instance Properties
    No_RefEngine : Integer;
    Engine_Type : String;
    Power : Real;
    Acceleration : Real;
}

Class CarBody
{ Class Properties
  Superclass : Object;
  Key : No_RefBody;
  Instance Properties
    No_RefBody : Integer;
    Body_Type : String; }

Create export Schema SchemaBody;
Correspondence C1
{ AssociatedClass : Car;
 ListOfAttribut : No_Ref,BodyCharacteristics, Option,Characteristics
  AssociatedPredicate : C1.No_Ref = Car.No_Ref }

Correspondence C2
{ AssociatedClass : CarBody;
  ListOfAttribut : No_RefBody, Body_Type;
  AssociatedPredicate : C2.No_RefBody = CarBody.No_RefBody; }

Create export Schema SchemaEngine;
Correspondence C3
{ AssociatedClass : Car;
  ListOfAttribut : No_Ref, EngineCharacteristics
  AssociatedPredicate : C3.No_Ref = Car.No_Ref + 5401; }

Correspondence C4
{ AssociatedClass : Engine;
  ListOfAttribut : No_RefEngine, Engine_Type, Power;
  AssociatedPredicate : C4. No_RefEngine = Engine.No_RefEngine; }

Figure 6 An ERM specification using MO-MER
6 A METHOD FOR THE ERM DESIGN

The ERM design method is divided into three processes: preparation, modelling and adjustment.

**Preparation** • Identification of the applications to integrate

**Modelling** • For each application modelling its data.
Taking into account new applications: studying its interaction with the connected applications

**Integration** • Integration of created schemas for the construction of a global schema

**Adjustment** • Establish the correspondences between global schema and export schemas

*Figure 7 An incremental method for the ERM design*

6.1 The Preparation

The preparation is the study of data exchanges between the different applications. It is a step of skimming through the manufacturing system applications in order to detect:
• the functional interaction between the applications,
• the needs in data exchanges,
• the shared data,
• operated data exchanges.

The different tasks to perform are:
• to establish a sketch of the data schema of each application. This sketch has to be
sufficiently detailed in such a way that the data manipulated by the application could be noticed.
- to study the intersection of applications data in order to detect shared data.
- to study the manually or automatically undertaken data exchanges.

6.2 The modelling

The modelling process of an application data in this method is divided into three steps:

- the definition of the classes: the number of classes used in a manufacturing system is generally very high. It is necessary to reduce this set of classes by referring to the area in which the application data evolve. It is also necessary to define for each class the attributes that characterise it by consulting the data manipulated by the application.

- the definition of the hierarchy of classes: it is to define the inheritance lattice of classes, i.e. to define the relationship classes/sub-classes. This relationship is identified by using the generalisation or the specialisation technique. The generalisation technique consists in considering each defined class as a sub-class and seeking the potential super-classes that verify the inheritance. The specialisation technique consists in considering each defined class as a super-class and seeking the potential sub-classes that verify the inheritance.

- the definition of the relationships between classes: the relationships between objects can be deduced only if the real world modelled is well understood, because data as given in files or on the classical database do not carry enough information to able the reconstruction of relationships.

6.3 The integration of data schemas of applications

The shared schema is obtained by integrating the applications data schemas. The integration of schemas is a technique used in database to construct a global conceptual schema from the merging of the different data schemas. The objective of the integration is to detect conflicts between the different schemas and to construct a conceptual schema without any redundancy. This objective is generally reached by defining a strategy of schema integration, by constructing the associate global schema and by verifying the validity of the obtained schema.

Many studies have been carried out to solve the problem of schema integration. Since 1977, many integration methods and many comparative studies have been published (Batini, 1986, Geller, 1992, Hayne, 1990, Lim, 1994, Spaccapietra, 1994, Suzuki, 1995). Each integration method has its own steps. Globally, these steps can be considered as a combination of the following steps:

- pre-integration: an analysis of schemas is undertaken to choose the strategy of integration to be adopted. Two types of integration strategies exist: binary or n-ry ones. Binary strategies consider the integration of schemas two by two, n-ry strategies consider the integration of any number of schemas simultaneously.

- comparison of schemas: schemas are compared to determine relationships between concepts and to detect conflicts that may exist.

- conflict resolution: once conflicts are detected, this phase creates compatible and therefore "integrable" schemas. Each conflict detected during the phase of comparison is eliminated by applying necessary transformations on schemas.

- merging of schemas: this phase constructs the global schema. It consists of a simple superposition of schemas if all conflicts have been resolved.
Part Three Integration Frameworks and Architectures

- the result verification: it allows us to test the quality of the global schema according to the completeness, the correction, the comprehension and the "minimality" criteria. The completeness guarantees that during the integration no information has been lost. The correction verifies that structures created by transformations in the global schema are valid according to the data rules of the model. The comprehension guarantees that defined structures are sufficiently explicit to be understood by users. Finally, the "minimality" guarantees that the global schema is exempt from redundancy.

6.4 The adjustment

In this step we complete the design of the ERM:

- by defining and by establishing correspondence between export schemas and the shared schema of an ERM,
- by detailing links between representations of same data in the different applications and in the shared schema.

This last point can be made by drawing up a table whose rows represent the exchanged data and columns indicate its representation in each application that hold it and in the shared schema. The figure 9 illustrates an example of this table.

<table>
<thead>
<tr>
<th>Data</th>
<th>Representation in the s/ERM</th>
<th>Representation in A</th>
<th>Representation in B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name</td>
<td>Type</td>
<td>Name</td>
</tr>
<tr>
<td>Temperature</td>
<td>T</td>
<td>Real</td>
<td>Temp</td>
</tr>
<tr>
<td>Volume</td>
<td>Vol</td>
<td>Real</td>
<td>VG</td>
</tr>
<tr>
<td>Pression</td>
<td>Press</td>
<td>Real</td>
<td>Pression</td>
</tr>
</tbody>
</table>

Figure 9 Data exchanged between the application A and the application B

At the end, we allocate to the modelled ERM to DEE and we parameter this last by:
- defining the initial data flow between the different applications,
- establishing the authorised user list and access rights of each.

7 CONCLUSION

DEE makes the applications in the manufacturing systems integrated applications with preserving them. This is a great advantage cause the existing applications are valuable acquisitions of the companies. The second advantage of the DEE is its openness in integrating new application, cause the applications are connected to the DEE through local interfaces that encapsulate the access to the applications data.

The MO-MER model has two advantages. The first one its similarity with the ODMG model that is considered as the future object oriented model standard. This will facilitate the integration
of new applications using object oriented data models. The second advantage is the reflection of
the MO-MER that has facilitated the communication between the DEE and the ERM. This one
offers information on itself and on the data it manages (ERM-Instances).

The design of the ERM can take a long time. The ERM design method presented here has been
oriented in a way to allow a progressive design of the ERM.

8 REFERENCES

Ahmed, R.; Smedt, P.D.; Du, W.; Kent, W.; Ketbachi, M.; Litwin, W. A.; Rafii, A. and
Shan, M.C. (1991) The pegasus heterogeneous multidatabase system, IEEE Computer,

Andersson, M.; Dupont, Y.; Spaccapietra, S.; Yetongnon, K.; Tresh, M. and Ye, H.
(1993), The FEMUS Approach in Building a Federated Multilingual Database System,
3rd International Workshop on Research Issues on Data Engineering : Interoperability in

Atrood, T ; Duhl, J ; Ferran, G; Loomis, M ; andWade, D (1993) The Object Database

BATINI, C ; LENZERINI, M ; NAVATHE, S.B. (1986) A Comparative Analysis of
Methodologies for Database Schema Integration, ACM Computer Surveys. Vol 18, No
8, 1986.

Exchanges, 2nd IEEE/ECLAl/FIP International Conference on Architectures and Design
Chapman and Hall.


Chung, C. W.(1990), DATAPLEX : An Access to Heterogeneous Distributed Database,

Clement, D.; Ganesh, M.; Hwang, S. Y.; Lim, E. P.; Mediratta, K.; Srivastava, J.; Stenoien,
J. and Yang, H.R. (1994), Myriad : Design and Implementation of Federated Database

Full and Partial Correspondence using the Dual Model, Information Systems Vol. 17, No.

System for View Integration, Proceeding of Int. Data Engineering, Los Angeles,
California, Février 1990.

Lim, E.; Srivastava, J.; Shekhar, S.(1994) Resolving Attribute Incompatibility in Database
1994.

Scholz-Reiter, B. (1992), CIM Interfaces, Concepts, standards and problems of interfaces in

Sheth, A.P. and Larson, J.A. (1990), Federated Database Systems for Managing Distributed,
Heterogeneous, and Autonomous Databases, ACM Computing Surveys, Vol. 3, No. 22,

conflicts., in IEEE Transaction on Data and Knowledge Engineering.

Conflict Resolution and Checking Conceptual Similarity -Conceptual Graphs Approach,
Proceedings of 6th Int. Hong Kong Database Workshop, Hong Kong, March 1995.

Wix, J. and McLelland, C. (1986), Data Exchange between Computer Systems in the
PART FOUR

Information Infrastructure and Business Processes
Value System Redesign: System oriented management and enterprise integration in globally distributed manufacturing networks

B. R. Katzy
Dr.-Ing. Dipl.-Kfm., research officer, lecturer
University St. Gallen, Institute of Technology Management,
Unterstrasse 22, 9000 St. Gallen, Switzerland,
Tel. (Fax): 0041-71-2282450 (420),
e-mail: Bernhard.Katzy@item.unisg.ch

B. Riggers
Dipl.-Wirtschafts-Ingenieur
University St. Gallen, Institute of Technology Management,
Unterstrasse 22, 9000 St. Gallen, Switzerland,
Tel. (Fax): 0041-71-2282450 (420),
e-mail: Bernd.Riggers@item.unisg.ch

Abstract
The paper will propose a way to augment „knowledge and appreciation of integrated systems“ for the purpose of the integration of globally distributed manufacturing sites. The objective is to provide an „associated methodology“ for managers in industrial enterprises, responsible for the design and operation of globally distributed manufacturing networks and their providers for business process re-engineering, ICT-infrastructures and logistics services. We introduce a strategic level of the enterprise system as a completion to the existing operative Enterprise Integration Architectures.

Keywords
CIMOSA, Enterprise integration, ICT-infrastructure, logistics, manufacturing, modelling, network, process re-engineering, strategy, TELEflow, value system, virtual
1 INTEGRATION IN THE IFIP/IFAC WORK

In the first Workshop on DIISM in Tokyo 1993, "a possible road map for promotion of the field of enterprise integration" (Williams 1993) has been defined. The paper will propose a way to augment "knowledge and appreciation of integrated systems" not for "potential users everywhere" but for the purpose of the integration of globally distributed manufacturing sites. The objective is to provide an "associated methodology" for that concerned "user application group". They are managers in industrial enterprises, responsible for the design and operation of globally distributed manufacturing networks and their providers for business process re-engineering, ICT-infrastructures and logistics services. We introduce a strategic level of the enterprise system as a completion to the existing operative Enterprise Integration Architectures.

2 THE PROJECT

This paper will present the concept and first findings of the TELEflow-project. The TELEflow-project has been selected as "flagship project" for the telematics application program in the 4th framework. Its objective is to develop methods and tools for business process re-engineering in global manufacturing networks applying advanced telematics technologies. It was officially started in February 1996. It has a duration of three years. Partners are ATM from Daimler Benz (D), Danzas (CH), GPS (D), Huber und Suhner (CH), Intracom (GR), Siemens Nixdorf (CH), TRD (GR) and University St. Gallen (CH).

3 THE CASE

The case is derived from one of the three pilot sites in the TELEflow-project where infrastructure services are designed and installed.

3.1 The situation

The company is a mediumsized, high quality provider of electrotechnical equipment for high-frequency installations such as antennas, HF-connectors and cables. The equipment is used for example in the basis stations for mobile telephone networks. The networks are provided e.g. by Motorola, Ericsson or Nokia who are customers. In 1995 turnover increased by 10,8 %, which was mainly carried by an increase of exports from 58 % to 67 % (N.N.(Gruppen) 1996). Another positive news for 1995 was the start of manufacturing in England, Canada and Singapore (N.N. (Ertrag) 1996).

3.2 The requirements

Existing linear models of business processes (figure 1, left side) that are focused on the customer and provided by distribution partners (A), manufacturing (B) and suppliers (C) have to be amended as to cope with the new business situation. First, because customers have to be served at globally distributed sites with the same quality, service and conditions.
Second as the business situation shows manufacturing is no longer concentrated to one site, instead now there are four sites (bullet B) to be regarded in the process chain. The linear value chain such changes to a network structure of involved partners (figure 1, right side). The partners can be different units of the company or independent business partners. We named this network a value system, as it is designed to create added value for the customer.

Business processes within the value system have to perform as efficient as those inside one corporation. As we focus on manufacturing industries logistics services are an important aspect of the value system, as its total cost today amounts to 10-25 % of the total product cost.
(Schuh 1996). Information and Communication Technologies (ICT) provide the third important aspect of the value system infrastructure.

Advanced technologies in all three disciplines, business process re-engineering, logistics and ICT, exist, but there is a need to harmonise those three independent systems in order to achieve the required efficiency (figure 2).

Systems theory is referred to in all three disciplines. In our experience this is as much a chance as it is a challenge! Today’s challenge is the very different application of the systems theory principles behind the same terms. „Behaviour“ for example that is modelled within IT-system design to control the dynamic invocation of functional operations for business managers describes pure „soft factors“ and is restricted to people. In fact misunderstandings are a major obstacle to the harmonisation of the different solutions. On the other hand the chance of a systems approach in our eyes lies in the integration of the existing solutions for a business application. But this requires a clear conception of the value system.

Industrial users like the case presented as well as service providers for ICT or logistics face the same need to understand the value system. The former as they have to design and operate their value system, the latter as they have to create and demonstrate the added value of their services as integrated parts of a value system.

4 THE VALUE SYSTEM

Tomorrow’s competition will no longer take place among single companies, but among value systems. A value system (VS) is the organisation of all companies (e.g. different sites, supplier, OEM, Distributor, Service Provider) that collaborate for the customer solution (figure 3).

Definition of the Value System:
A Value System is a focused network of companies and units, in order to maximize the added value created, by optimizing the intercompany value chains.

Figure 3    Definition value system
1) The strategic level of the value system

This definition deliberately includes the „what is the value system for?“. This is to point out that the value system describes the application technologies and the allocation of people or organisations in a competitive environment. Generating superior value for the stakeholders is the reason for the value system to exist and the entire network is focused on this. Necessary elements of the entirely human oriented strategic level of the value system are derived from the analysis of what is necessary to achieve superior value for the customer.

2) The operative level of the value system

„What does the value system look like?“ is the description of the infrastructure of the network throughout the system life cycle in terms of requirements that it has to fulfil, its design and implementation.

3) Integrated management of the value system

A value system is a consortium of interacting, legally independent companies and units. Value systems have to be managed to sustain competitiveness. They have to be designed, operated and continuously improved in dynamic environments. This means to broaden the domain of interest for manufacturing management and to actively re-engineer inter-company-processes of the value system. Methodologies are needed for this value system management of a fluid or flexible network organisation in a highly dynamic environment. The management of such a Value System faces the challenge to gain a fit between operative and strategic level.

5 THE VALUE SYSTEM FRAMEWORK

5.1 Dimensions

The framework has been defined to find a linking structure across the traditional three views and the strategic and operative level. It is intended to provide a concept that allows managers (1) to structure the complexity of the VS, (2) to understand their roles in the VS and (3) to support pro-active decision making in the VS. Three relevant dimensions of the VS framework¹ have been fixed to cope with the needs of the VS management (figure 4):

¹ in analogy to Bleicher (1995)
As the challenge of competitive VS will be the integration of geographically distributed sites and companies (as a specific form of a virtual organisation) the distributed structure of this network is a first important dimension of the value system. Where the distributed structure is predominant for the providers of infrastructures (i.e. the ICT components) harmonised activities of the value system such as successful order processing or harmonised marketing programs are predominant for the industrial user. Global value systems require the collaboration of people from different companies, different countries and the three disciplines. The behaviour in a multicultural context has been stated by managers as one of the most critical success factors for the performance of value systems.

As this conference focuses on information infrastructures that we take as a part of the distributed structure of the value system I will step into detail on this first dimension, keeping in mind that there are close interdependencies with the two others.

5.2 Profiles as a strategic description of the value system

*Systems design as operative implementation of a value system*

Architectures for the integration of manufacturing enterprises provide a broad variety of constructs to describe different components of the distributed structure of a value system. Among others these are the Information System (ARIS) (Scheer 1991), the Decision System (GRAI) (Dourinings 1987), the Organisation (CIMOSA) (Zimmermann 1995), Business Processes (ARIS, CIMOSA) and communication (CORBA) (Ben-Natan 1995). It is our intention to use these constructs to model the infrastructure components of the value system.

*Business concept or strategic design of a value system*

But we found that managers in value systems face the problem to integrate and harmonise existing (and mostly diverging) structures inside the partner companies. As it is normally impossible (to expensive or takes too much time) to implement new structures from scratch, it is necessary to find similarities between the existing structures which serve as leitmotivs to integrate them in a value system. Or, it can be foreseen that an integration may probably not produce synergies. We call these key characteristics the value system profiles (fig. 5). From interviews we identified the four given aspects concerning the distributed structure of a value system. They summarise the main perception of a Value System by managers in charge. We
use portfolio techniques to visualise the profile, because managers are accustomed to it. This top down profile can be refined by a set of detailed criteria. Managers have perceived the information policy (lower left corner) within their value system between the extremes fully „transparent“, where information is accessible to every employee, or „task oriented“, where information is used as an instrument of power, on the other side. Certainly there are many forms in-between the extremes that we currently describe by examples and statements. We don’t use figures to let the value system be described (50% transparent, 50% task oriented), because they are unsuitable and not meaningful enough. The description of such a profile requires many interviews, so that this refinement has to be seen as an ongoing process.

![Figure 5](image)

**Figure 5** Profiles for value systems: Example for distributed structure

Industrial users may use this to assess the situation of their value system, identify the conflicts and derive the requirements for actions to be taken and structures to be implemented.

6 SYSTEM INTEGRATION OF THE VALUE SYSTEM USING ENTERPRISE MODELLING TECHNIQUES

Value system management can be supported by enterprise modelling techniques. The framework has been designed as a complement to the requirements definition level to extend the system life cycle into „earlier“ phases of business integration of the strategic level of the VS. As our first experiences show the industrial applicability of enterprise modelling techniques has been augmented with this framework. In our experience this is due to the fact that with this framework industrial managers are addressed as an own user group.
We feel encouraged to this work as we interpret several indications within existing enterprise integration architectures as the expression of a need for such frameworks. Among others these are the „Enterprise View“ within ODP-CORBA (Ben-Natan 1995), the installation of a „Business Object Management Special Interest Group“ within OMG (Wagner 1996) or the recommended work on „enterprise integration methodologies“ with CIMOSA (Kosanke 1995). But this actual work still needs a clear reference point for the development of solution. For the value system framework we decided to search for this solution from a strict application point of view. Mature technologies will be looked at from this application point of view.

In order to deal with the differences to the existing enterprise modelling architectures intend to keep the specifications of the value system framework as a distinct „strategic level“ (fig. 6).

Figure 6 Strategic level and enterprise integration architectures

Differences in Modelling
A first difference of the strategic level is the focus on the contents rather than on the formal correctness of the constructs applied. Consistency is measured by the credibility of the conception of the value system.

A second difference we discovered was the indicative character of the strategic level in contrary to the descriptive character of most existing enterprise modelling architectures. The objective is to create a shared conception of the value system between the business managers involved. This difference leads e.g. to a different definition of what is complete. The strategic level is regarded as complete when nothing can be skipped without losing the essential idea of the concept. Additional details in the best case are regarded as superfluous, mostly as misleading and often as tutelage. As most partners volunteer to participate in the value system they have to be motivated by a strong idea of the value system to bring in their own decentral solutions for the common purpose. Detailed centralised value system models in this sense are dysfunctional as they are perceived as an attempt for order and control.
Object oriented techniques fulfil the requirements of decentral co-ordination. For „managed objects“ it can be negotiated (for all people, organisational units or infrastructure components) what is public in the value system and what remains private. On the other side the object oriented approach allows for the integration with existing modelling techniques.

Other time-tested concepts from enterprise integration like the instantiation process will be kept for the operative level. A „generic“ value system as a general conception of what a value system is, will be described to step forward to best practice or benchmark descriptions as „partial models“ and to provide for the design of individual or „particular“ value systems during the work of the TELEflow project. Based on the application of existing methods and tools the integration concept for the value systems is enhanced by three operative instruments. First, a process reengineering method for inter-company processes in value systems is developed. Second, a method to map business process specifications into the communication system architecture is adopted from CORBA. Third, on the basis of stable business process modelling of enhanced logistics solutions will be supported.

7 CONCLUSIONS

Based on existing technical solutions in the three disciplines business process re-engineering, logistics and ICT the presented user driven concept of the value system has been developed to guide the integration of value systems in the TELEflow-project. Following the expressed business needs of the industrial partners existing enterprise modelling architectures (e.g. CIMOSA for business process modelling and CORBA for the design of communication transparency platforms) have been enhanced by a „strategic level“ to express the conception of the value system and bridge the gap to the (technical) system life cycles. Even in this early stage of the work this has proven to be an important „interpreter“ between „user needs“ and „technical providers“.

8 REFERENCES


---

2 cf the model instantiation process, Esprit Consortium AMICE (1993)

9 BIOGRAPHY

Dr.-Ing. Dipl.-Kfm. Bernhard R. Katzy
born on April 9th, 1962 in Aachen
started his professional career as a car mechanic. Later he studied Electronics and Business Management at the Technical University of Aachen.
After his PD on „Modelbased Enterprise Engineering for Manufacturing“ he joined the Institute of Technology Management in 1994 as Research Officer and Lecturer at the University St. Gallen.

Dipl.-Wirtschafts-Ingenieur Bernd Riggers
born on December 12th, 1965 in Hamburg, Germany
after finishing his studies in business administration and mechanical engineering at the Technical University Berlin in 1993 he worked for a consulting company in the fields of system analysis and business process reengineering. In October 1993 he moved to St. Gallen to start his PhD-Thesis, in the field of agile manufacturing networks and value system redesign, at the Institute for Technology Management at the University of St. Gallen, Switzerland. Actually he is project leader of the European research project TELEflow, which has the objective to provide companies with a holistic management concept for designing and operating interorganisational networks.
Confidence in a Network of Cooperating Companies
-- QM-Documentation for Certification

K. Mertins, M. Schwermer, F.-W. Jäkel
Fraunhofer Institute of Production Systems and Design Technology (IPK) Berlin,
Division of Systems Planning
Pascalstrasse 8-9, 10587 Berlin, Germany
Telephone: ++49 30/39006-164
Telefax: ++49 30/3911037
E-Mail: frank-walter.jaekel@ipk.fhg.de

Abstract
Cooperation between and within enterprises should be strengthened. Transparency of the processes promotes communication and confidence between all partners. International cooperation calls for precise sequences of operation as well as for performances in line with market conditions, i.e. performances that are available on time and in the necessary quality. Modeling leads to a clear view on business processes and is an essential step in the process of organizing enterprises. Relevant aspects of modeling and discussion contain all data about the quality of processes and the quality of their performance. It could also be used as basis for QM and organizational manuals. Integrated Enterprise Modeling (IEM) represents planning information transparently and is therefore the basis for discussion between project participants. In order to evaluate the variety of planning information and description requirements IEM enables different views on one consistent model as well as the business processes belonging to them. The software tool MO²GO (method of object oriented business process optimization) supports the modeling process based on the IEM method. It allows a transparent description of the business processes as well as an analysis of costs, times or quality to enable participants to identify optimization potentials.

Keywords
Object-oriented, business process, IEM, MO²GO, ISO9000ff, enterprise modeling, process modeling, business process modeling, business process analysis, business process reengineering, reengineering quality manual, QM-documentation, certification
1 INTRODUCTION

Worldwide competition is getting fiercer every day and requires companies to strictly orient their operations towards the market. Constantly changing markets require a high speed of response and a high degree of flexibility. To survive in a competitive climate, companies also need to place great emphasis on customer orientation. Cost pressure and customer orientation require continuous improvements in the quality and efficiency of business processes and quick adjustments to shifts in the market.

One way to increase the flexibility and reduce costs is to introduce decentralized, self-reliable forms of organization. The use of worldwide communication networks (e.g. Internet) invalidates the importance of proximity within a decentralized company. Independent companies with locations all over the world can therefore cooperate and divide their labor.

Different customer specific demands leads to changing partners for cooperation. The identification and cooperation with new partners requires some trust in their process and product quality. Quality management and business process modeling are already available in many companies. The development of methods to optimize business processes remains to be one of the essential tasks of corporate planning. Furthermore, in view of dynamic markets and changing basic conditions such as laws and the policy environment users have to be able to analyze and adjust business processes on an ongoing basis. Methods need to be developed that allow users to continuously use business process models and the appropriate tools.

International cooperation calls for precise sequences of operation as well as for performances in line with market conditions, i.e. performances that are available on time and in the necessary quality. In the course of international job-sharing efforts between different independent companies, confidence in the productivity of the respective companies is gaining growing significance. The processes leading to and the quality of the performance are equally important. Especially the quality of the processes may be established with the corresponding documentation and subsequent certification (e.g. according to ISO 9000ff). The business process can be analyzed and improved simultaneously. The objective is to develop transparent and up-to-date documentation to insure the necessary communication between the people involved and to identify the suitable partners.

2 METHOD

Information on the quality, control, optimization, costs and the comparison (process benchmarking) of processes as well as on responsibilities and environmental aspects may be obtained with different methods. The required data and structures, such as models of the organizational structure and the process organization, are often similar or are based on similar basic structures. A method to model and analyze business processes needs to take these different model views into consideration. The method must also permit users to employ the recorded data and models beyond currently studied partial tasks. The objective is to introduce process models as a basis to easily record and process information, to describe the efficiency and to insure the performance of the company beyond its boundaries.

Business Process Reengineering appears to be a way to develop the most successful customer-oriented business processes. To optimize business processes it is necessary to examine and plan several aspects of a company (GRO94). These aspects are illustrated in figure 1. Modeling methods used for BPR must elucidate these aspects and their interrelations.
A model core should allow users to describe changes, for example regarding the organizational structure. The effects on processes and throughput times have to become visible. To evoke the participation of management and staff the models have to be easy to create and understand.

The derivation of different views on the modeled information should be possible. These views should filter the information that is then focused in new structures and layouts. An important view is the documentation of QM systems. It is therefore necessary to describe the aspects that were defined by ISO 9000 ff and to transfer them into the layout of a QM manual. The derivation of a QM documentation from an enterprise model results in a documentation that is consistent and easy to update.

In this way modeling methods could support the introduction or optimization of quality management as well as the development and update of quality manuals. The use of the same models to document business processes and quality manuals enables users to automate the update of the manual with every process innovation (MER95b).

Figure 1 Requirements of Modeling Methods

To fulfill all the different requirements an object-oriented concept is used (COA90, HAM94, RUM91). The method takes advantage of the object-oriented approach to integrally describe information and functions as views on a single model of the company. The core of the model structure includes the views ‘business process model’ and ‘information model’. The production and all activities connected with the production are described in the model as functions and business processes related to objects (MER95a, MER95b). The concept is based on the method of „Integrated Enterprise Modeling“ IEM (SPU93, MER95c). When constructing the model the required data and functions are related to objects (SPU93). The relations between the objects are determined. According to a selective level of detail this
results in a comprehensive registration of the tasks, the process organization, the corporate data, the manufacturing system and the components of the information system.

In this context, business processes are understood as chains of operational activities and their network-like relations that are oriented towards the objects 'product', 'order' and 'resources'. Between the business processes is a customer-supplier-like relation. They extend over organizational and system boundaries and have a defined overall result. On the basis of this relation, one may develop network-like structures with different partners.

Customers are mainly interested in the provision of a service, less in how this service was provided. However, customers have to trust the performance of the supplier. The quality of services and products according to the schedules are therefore of crucial importance. QM documented by business process modeling is a way to achieve these confidence.

The modeling approach permits the description of a network of processes along with the respective companies as well as the description of the processes of the own company. The described method is suitable for many planning and structuring tasks in companies. The application includes the design of material and information flows. The systematic and transparent description of business processes as a communication base between the departments and between the different hierarchical levels proved to be successful in several projects. Among other things, time saving potentials were made clear. The distribution of costs was improved with regard to the respective 'initiators' and the deployment of personnel was improved with regard to qualifications.

The exchangeability of models between project groups and companies requires a uniform modeling language with standardized constructs. The modeling concepts presented here coincide with the 'Framework for Modeling' (ISO TC 184/SC 5 WG 1). The basic language constructs are part of the European pre-standard 'Computer Integrated Manufacturing- Constructs for Enterprise Modeling’ ENV 12204.

3 DERIVATION

The object-oriented approach enables the user to relate different process analyses to a model core. Models and information required for one analysis need only be supplemented for further analyses (COA90). They do not anymore need to be constructed anew.

For business process reengineering purposes the model is used to analyze weak points and to identify improvement potentials. For that purpose the corporate structures may be visualized, reference models may be employed, times and costs may be analyzed and model-based discussions may be carried out.

IBM enables users to locate and visualize improvement potentials on the basis of the model structure. For example, interruptions of the process support by information systems and interfaces between organizational units can be located and visualized. Order processing and product development processes, construction processes and resource processes were studied and optimized in industrial projects.

Figure 2 shows the most important aspects to evaluate such processes. To model these aspects means to have the basis for a documentation of the Quality Management. It can be used as information for all partners in a network about the functional capacities and the QM of the enterprise.
### Important aspects for enterprise modeling.

A process model that was developed in a reengineering process can be used to create manuals (QM manual ISO 9000ff, Environmental Management Manual ISO14000) by way of adding additional classes and attributes (MER95b). A corresponding class structure for QM manuals has been developed at IPK Berlin. It has already been applied in projects. Two ways are possible to come up with enterprise specific models for automated generation of QM manuals. One may either use and adapt existing enterprise models or develop new ones. IPK Berlin has also developed branch-specific reference models for the food industry and for software suppliers. These reference models further reduce modeling expenses and facilitate the creation of manuals for the certification of companies.

The relevant information may be described as part of the process model, as attributes of the object structures or as textual descriptions of objects. The structures of the QM-manual, the general orders and the work assignments are defined as resources. Each chapter and each instruction is specified individually and is supplemented by textual descriptions. The usage of the general orders and work assignments, of other documents, devices and responsibilities is described in the process model.

When creating the manual, additional information is linked with the process model (figure 3). The information can be supplied directly by a graphic description of the process model or indirectly by appropriate documents that were generated by the model. The possibility to transparently describe business processes and to provide additional information leads to further possible applications of the business process model in the company.

<table>
<thead>
<tr>
<th>Aspects to model</th>
<th>Why?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processes</td>
<td>is the kernel of planning task; thinking in processes should be improved against identification with departments</td>
</tr>
<tr>
<td>executing organizational units</td>
<td>identification of organizational breaks and potentials for cooperation</td>
</tr>
<tr>
<td>Responsibilities</td>
<td>necessary for QM-Manuals</td>
</tr>
<tr>
<td>Data Documents and EDP-Systems</td>
<td>to show sources and needs of informations; breaks between EDP-Systems</td>
</tr>
<tr>
<td>Time</td>
<td>important aspect for process optimization (Throughtput-times) and project-management</td>
</tr>
<tr>
<td>Costs</td>
<td>one of the main aspects for evaluation of processes</td>
</tr>
<tr>
<td>operating devices</td>
<td>necessary for QM-Manuals</td>
</tr>
<tr>
<td>(e.g. measuring tools)</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 2**  Important aspects for enterprise modeling.
A key point is the continuous analysis and optimization of business processes. This may be done by employees who are adequately informed as well as by optimization measures such as process benchmarking.

For this purpose, the IEM model may also be used by way of adding process index numbers (attributes). These are required to evaluate processes and to compare different processes. Therefore, the model not only offers structural information for processes, but also explicit index numbers. The attributes and their values permit the use of the model as an information platform as well as the evaluation and comparison of sequences of processes.

Figure 3 Automatic Derivation of Documents from the Process Model

A selection of the various possible applications of the IEM enterprise model was presented. The simultaneous execution of different activities may be sensible and may also reduce costs. The creation of a QM manual, for example, could be connected with a study and optimization of existing business processes.

4 FURTHER DEVELOPMENT

The various possibilities of application of IEM business process models presented thus far have all been tested and have mostly been employed in industrial projects. In the future, the integration of business process models and business process analyses needs to be emphasized. Ultimately, the business processes and tools have to be continuously adapted to the ever-changing environment. Therefore, the employees have to be integrated into the business processes, for example through supporting workflow systems. Important prerequisites include the possibility to exchange data between different software systems (PPC, business process
Cooperating companies and QM-documentation

modeling tools, workflow and information systems) and the appropriate connection between the companies.

The data exchange between software systems within the company and tools to model and optimize business processes enables users to continuously analyze, inspect and control the business. Constant expenses for modeling and data input are reduced or become void. Therefore, the determination of the processes' need to adjust on the basis of pre-defined critical values is enabled.

The connection of corporate data with the business process model may be based on existing or currently developed standards. Product specifications can be exchanged through STEP (Standard for the Exchange of Product Model Data; ISO 10303). A standard for the exchange of data through processes and proceedings is currently being developed as STEP, part 49 (Process Structure and Properties Model). Corresponding standardization studies are carried out, among others, by the Workflow Management Coalition.

5 TOOLS

Tools are available to effectively apply the object-oriented modeling to business process modeling and analyzing.

As a tool to develop a model-based QM documentation the user is equipped with a guide. The guide includes the description of procedures of certification in conjunction with the necessary steps to develop a company-specific model. The QM documentation can be derived automatically from this model. The guide includes the description of the following steps:

1. Realization of a preparatory workshop - the goals and sectors of the company relevant for certification are determined, a job schedule for further activities is set up.
2. Development of an aggregate planning concept of the QM system - the enterprise model is developed, improvement potentials of the business processes and the QM system are identified: cost-benefits are estimated, a plan to introduce or extend the QM system is established.
3. Initiation of a QM team - a project team is set up and is entrusted with the task of introducing the QM system.
4. Development and realization of the QM system - the QM elements are defined and, afterwards, described in the enterprise model; measures to optimize the business processes are developed and enacted.
5. Introduction of the QM system - the employees are informed and trained; the QM system is 'exemplified' by selected promoters.
6. Certification - an internal pre-audit and the certification audit are carried out.
7. Stabilization, QM coaching - the QM system is improved continuously and is documented and prepared for following certification activities.

Libraries containing examples and reference models to design models effectively and quickly are supplied. System-specific reference models support the selection and introduction of standard software. Classes for analyses and specific applications, examples of various industrial projects and reference models for specific fields of application, for example the production of unique items or the creation of QM manuals, are available. A support system for the application of reference models and their dynamic adjustment is being developed. In the
future, they have to support and speed up the development of company-specific (reference) models through appropriate tools. For that purpose, certain procedures are supported, for example the introduction of decentralize structures. The construction of the model will then take place interactively in a dialogue with the user.

Many of the applications and fields of application described in this paper, such as the automatic generation of manuals, can only be employed successfully with the support of appropriate software. A suitable software tool has been developed.

The software tool MO\(^2\)GO (figure 4) supports object-oriented modeling with IEM. The universal tool for the description and analysis of operational structures and business processes allows the comfortable description and the purposive analysis of products, orders, resources and the respective business processes. The main advantages of the application of the tool are the possibilities to systematize the reengineering and optimization process and to reuse the enterprise model for later projects with different objectives and optimization tasks.

With MO\(^2\)GO the design and certification of companies are sped up and facilitated. MO\(^2\)GO increases the acceptance by employees and thus reduces costs. Refinement functions, deposited modeling rules and structured procedures support a structured approach to modeling tasks. The possibilities of the object-oriented approach and extensive functions to define objects and business processes enable the user to describe his specific notion of the company.

The illustration of the results of reengineering and optimization processes is supported by various evaluation and documentation functions. Results may therefore enter the operational decision-making processes much faster.

Additional classes and attributes can be added to MO\(^2\)GO models easily. These would then be available in libraries and could be loaded from there. For example, corresponding class structures to generate ISO9000ff documents could be loaded into existing models. With the help of these classes, QM manuals, based on business process models, can be developed automatically (figure 3). The maintenance of the manuals is facilitated if changes within the company are directly documented in the models. The design of business processes and the development and actualization of QM manuals are being integrated into one operation.

To support standardization efforts within the framework of STEP an EXPRESS (IND92a, IND92b) interface was specified for MO\(^2\)GO. The interface, as a connection between the process model and the product model, may be further developed and may then be implemented. MO\(^2\)GO includes a method to connect the process and the product model. The description of the product structure is connected with the process model through object relations and stages of the product life cycle (figure 4).

Macros to develop specific display formats of results are available, for example to automatically generate formatted MS-Word for Windows documents or EXCEL graphics. These techniques allow the statistical analysis of cost and time attributes of the model as well as their graphic representation. Statements on order and resource relations as well as on the creation and the use of documents can also be made. Target performance comparisons or benchmarking may use these index numbers of different models.

6 Practical Experiences in a Few Words

The method is used in manufacturing sectors, service sectors and distribution areas. It enables users to design and analyze integrated enterprise models. Different companies may be involved
in the individual processes. An example is the integration along the process sequence product development, production, shipping, maintenance and recycling.

Experiences using the method and the tool for the automatic generation of QM manuals on the basis of a business process model were made in the food and software industries. The experiences revealed that customers prefer certified contractors. But why do they want their suppliers to be certified and what are the benefits of certification really? These questions need to be answered because the quality of certificates still varies!

Figure 4  Process Model and Product Structure.

7  CONCLUSION

Integrated Enterprise Modeling (IEM) enables users to describe the company-wide control, production and resource provision processes along with the required data in one integrated model. The description of distributed structures (networks) can take place with the same method as the description of the individual processes or performers. The object-oriented method was developed in compliance with the requirements of industrial projects and
standardization efforts. The method was tried and tested in a multitude of operational applications. It enables users to parametrize processes easily and describes the life cycles of the objects changed by the processes, e.g. product life cycles. IEM is supported by a modeling tool for the object-oriented business process optimization (MOGO). The transparent description of the business processes, supported by this method, serves as an effective basis for discussions and enables users to identify optimization potentials on the basis of the model structure. The object-oriented method may serve as basis for workflow analyses, the calculation of process costs, benchmarking and simulation. Furthermore, it could be used to specify and implement workflow systems.

The approach reduces the expenses to register the required data and, in connection with a software tool, also the expenses to develop specific documents (such as the ISO 9000 ff documentation). The utilization of the model in the company is facilitated as well.

The method leads to cost effective documentation of business processes and QM systems. The process models specify possibilities and requirements for the cooperation of companies. To prepare these transparent descriptions of processes and used QM methods for possible partners cooperation means to start an honestly and open partnership. It is an effective way to achieve confidence in a dynamic network of cooperating companies.

8 REFERENCES


Cooperating companies and QM-documentation


9 BIOGRAPHY

Dr.-Ing. Kai Mertins was born 1947 in the Federal Republic of Germany. After studying Control theory in Hamburg and Economy together with production technology at the Technical University of Berlin, he became member of the scientific staff of the University Institute for Machine Tool and manufacturing Technology (IWF), Berlin/FRG.

Since 1983 he had been head of the department "Production Control and Manufacturing Systems" at the Fraunhofer-Institute for Production Systems and Design Technology (IPK), Berlin/FRG, where he has been Director for Planning Technology (President: Prof. Dr. h.c. mult. Dr.-Ing. G. Spur) since 1988.

He has more than 15 years experience in design, planning, simulation and control of flexible manufacturing systems (FMS), manufacturing control systems (MCS), shop floor control systems, and computer integrated manufacturing (CIM). He was General Project Manager in several international industrial projects and gave lectures and seminars at the Technical University Berlin, PolytechnicNottingham/UK, Czech Republic, Indonesia and China.

Special field of interest: Manufacturing strategy development, planning for production systems, shop floor control and simulation.

Dipl.-Ing. Martin Schwermer was born in 1963, studied economical engineering at the university of technology in Berlin. Since 1988 he is a member of the staff of the Fraunhofer Institute for Production Systems and Design Technology (IPK Berlin), department of systems planning. He was Project Manager and Project Leader in several industrial projects in the field of production and process planning.

Dipl.-Ing. Dipl. Inform. F.-W. Jäkel was born 1959 in Germany. After studying vehicle design in Hamburg and computer science at the Technical University of Berlin. Since 1992 he is a member of the staff of the Fraunhofer Institute for Production Systems and Design Technology (IPK Berlin), department of systems planning.
Applying Information Technology to Minimise Risks in Satisfying Organisational Needs

O. J. Akomode¹; B. Lees¹; C. Irgens²
University of Paisley,
¹Department of Computing and Information Systems,
²Department of Mechanical and Manufacturing Engineering and The Quality Centre.
Paisley PA1 2BE, Scotland, UK.
Tel: +44-141-848 3327 Fax: +44-141 848 3542
Email: akom-ci0@paisley.ac.uk

Abstract
The profound effect of Information Technology (IT) is changing the way managers of most organisations carry out their activities. In organisational management IT has the potential to offer accurate and timely data/information to support decision making. But research reveals that there are various types of risk associated with the development and successful integration of IT-based Information Systems (ISs) at organisational level. In attempting to meet business needs or organisational goals the situation often leads to a plethora of problems. The failure of some organisational IT-based information systems are reviewed and possible types of risk associated with the application of IT in meeting organisational requirements are discussed. Based on the experience gained in an ongoing project for developing an IT-based IS for manufacturing management, an anti-positivist methodology (Action Research) is proposed, which articulates a ‘learning’ process for both the business ‘owners’ and the IT analyst/developer, as a suitable means for eliciting the required knowledge. It is anticipated that the discussion presented may help to generate and increase an awareness of risks of failure when combining IT with organisational needs, so that such risks can be minimised or avoided through the suggested approach.

Keywords
Information Technology; Information Systems; Risks; Anti-positivism; Action research; Knowledge elicitation.

1 BACKGROUND

The creation of an efficient and effective manufacturing industry may be a means of assisting the improvement of people’s standard of living. Equally, the commercial life of an organisation depends upon the customers buying its products and services. Various new
information and communication technologies are helping to create new opportunities and secure a competitive edge for most organisations. In manufacturing management Information Technology (IT) has the potential to help in creating useful information infrastructure systems to support managers in decision making, in order to minimise (or entirely avoid) business risks in manufacturing operations. But research findings reveal that all projects which involve the development of IT-based information systems (ISs) exhibit some form of risks to the organisation. The point could equally be substantiated by the remark of Willcocks and Margetts (1994, p.128) that Risk is involved in all IS projects. Inadequate assessment and management of risks in the development of a computer-based IS in an organisation may lead to failures in the successful performance of the information system, which in turn may lead to other business problems. To drive home the point about failures of organisational ISs, examples could be cited of major instances in the UK such as: 

(a) in 1991 a commercial bank (National Westminster) abandoned a £20 million IBM DB2 Share registration system that was to link the bank with the London Stock Exchange’s Transfer Automated Registration of Uncertified Stock (Taurus). The abandonment of the Share registration system was said to be due to lack of adequate specifications for the trading system Taurus (Computing, 16 May 1991, p.1); 

(b) Taurus itself was one of the greatest failures of bespoke information system developments, costing a total of £400 million which was aimed at being a paperless Stock Exchange system but ended up generating more paper than was ever dreamed of and was eventually abandoned (Computing, 25 March 1993, p.52). The implication of the two examples is that while the failures of major information system projects (such as the NatWest IBM DB2 Share registration system and Taurus itself) may be loudly reported in the news media, there could be many others that fail in less noticeable or less publicised ways.

A distinction could be drawn between IT and IS, in that: 

(a) IT represents the competence presented by computer hardware, software applications and telecommunication technologies; 

(b) IS represents a wider notion which encompasses various intelligence gathering devices put together to meet the defined information requirements of an individual (or an organisation) in attempt to properly control the surrounding. IS may or may not be IT-based (cf. Davenport and Short, 1990, p.11; Stowell, 1991, p.174; Willcocks and Margetts, 1994, p.128). In what follows the term IS will be assumed to include the potential offered by IT. Details presented are based on findings from an ongoing project at the University of Paisley, into IT-based risk assessment decision support for the tendering process in manufacturing management, in collaboration with Renfrewshire Enterprise, and Compaq Computer Manufacturing Limited, Bishopton, Scotland.

2 TYPES OF RISK IN ISs AND BUSINESS RISK FACTORS

Risk could be broadly characterised as the possibility of a negative outcome and the consequences of that possibility (cf. Hertz and Thomas, 1983, p.3; Brauers, 1986, p.139). Risk management constitutes a practice of reacting to perceived risk by some form of assessment or observations in order to reduce (or entirely avoid) the unfavourable consequences that may ensue should the risk occur. In the development and implementation of ISs for organisations, types of risk that could be encountered may include: (i) extended budgetary cost, due to over-stepping the amount initially allotted for completing the
project; (ii) longer time for implementation; (iii) inadequate systems specifications, which may be due to lack of proper understanding of the business needs by the IS analyst(s)/developer(s); (iv) poor performance of technical systems, which may be due to choosing unsuitable hardware/software for the business system; (v) inadequate data model, which may due to the systems analyst/developer not obtaining sufficient (or appropriate) business data to model and modify the knowledge-base of a required Knowledge Based System (KBS) for the IS; (vi) incompatibility of the system with other information systems of the organisation; (vii) failure to achieve some (or all) of the expected benefits due to users’ ill-understanding of operational techniques or other implementation obstacles. Furthermore, these types of risk in IS may have an adverse impact on an organisation’s effectiveness and efficiency to profitably satisfy its customers.

Based on theoretical and empirical investigations in the current research project of developing a prototype IS for the tendering process in manufacturing management, the possible types of risk associated with the development and implementation of ISs as listed above could further have an impact on and compound an organisation’s business risk factors. In manufacturing management, results obtained from organisational investigation indicate that the risk factors which are often considered in practice are both quantitative and qualitative, encompassing the areas of: (i) total cost/benefit assessment in monetary terms; (ii) quality in terms of fitness for purpose; (iii) technology advantage; (iv) price and profitability; (v) timely delivery of products/services; (vi) image attainment and its sustainability; (vii) long term partnership relations and its proper management (with suppliers and customers) in terms of shared business risks and shared rewards; (viii) safety. These risk factors may also be applicable to the service industry. The types of risk and business risk factors aforementioned are not exhaustive but they serve to illustrate the potential risks in ISs. The exposure of an organisation to risks in ISs may increasingly become prominent when such risks further affect parameters of its ‘business deliverables’ to customers and other stakeholders. In extreme cases the commercial viability of the company may be seriously jeopardised.

Proponents and exponents of risk assessment/analysis of IS projects in organisations have evolved some models to help in evaluating various possible types of risk at the feasibility stage in order to avoid pitfalls. For instance, Corder (1989, pp.242-4) discusses the strategic weighting of risk factors in estimating computer projects, and presents a table for the calculation of strategic risks associated with such projects. The method identifies risk factors in organisational ISs and classifies them into three groups specified as: (1) high-risk factors, encompassing the five components (a) project size, (b) project definition, (c) user commitment and stability, (d) elapsed time and (e) number of systems interfaces; (2) medium-risk factors, which includes the seven elements (a) functional complexity, (b) number of user department, (c) newness of technology/vendor, (d) user experience of computers (e) the project team’s experience of the user area, (f) newness of technology to the organisation, (g) number of vendor/contractors; (3) low-risk factors, covering the three elements (a) number of sites, (b) functional newness (c) number of project phases. Some other models include that of Parker et al (1988) and that of Cash et al (1992; also see Willcocks and Margetts, 1994, p.128). While these approaches may be useful they are likely to fall short of offering a ‘complete solution’ to risks reduction (or avoidance) in the development and successful implementation of organisational ISs; they tend to lay emphasis
Minimising risks in satisfying organisational needs

on the feasibility (or initial) stage. But the initial stage which represents only a part of a coherent ‘whole’ in an IS project may be largely based on financial and statistical evaluation techniques that do not fully consider the human and business implications. Therefore, a problem-solving methodology which articulates an iterative learning process for both the organisational participants and the ISs analyst(s)/developer(s), then considers the various stages of the project may have a better potential in helping to reduce (or entirely avoid) the risks associated with all the stages (e.g. feasibility, design, development, implementation, training and use). The methodology suggested here and adopted in the project is the anti-positivist paradigm of social (or organisational) inquiry and analysis which is further discussed in the sections below.

3 POSITIVIST AND ANTI-POSITIVIST APPROACHES TO ISs

The philosophies of positivism and anti-positivism in organisational inquiry draw upon the assumptions of conceptualising the nature of science by subjective - objective dimensions for social inquiry; while the assumptions about the nature of society can be thought of as regulation - radical change dimensions (Burrell and Morgan, 1994, pp.21-23). The positivist (or ‘functionalist’) perspective involves the application of models based on natural science (such as in physics, engineering or biological methods) to the study of human socio-cultural affairs and organisational analysis (ibid pp.25-28). In terms of the development and implementation of an IS the implication is that the systems analyst/developer plays the explicit role of an observer of actions. Soft Systems Methodology (SSM) clearly points out a breakdown in the application of the natural science approach to a situation of problem-solving in social (or organisational) inquiry and analysis. SSM suggests an implicit participation and articulates a learning approach to organisational inquiry and analysis (Checkland, 1981; Checkland and Scholes, 1990).

In the anti-positivist (or ‘interpretive’) approach to organisational investigation the researcher (or analyst/designer) is an active participant in the process with the relevant group in the organisation. This contrasts with the natural science approach in which the researcher (or analyst/designer) is an observer, external to the process. The concept (based on the philosophy of SSM) seeks individual consciousness and human participation in a problem situation as opposed to that of an observer of action. The idea could be said to be of basic meaning that underlies social life (cf. Burrell and Morgan, 1994, p.31). With regard to information systems design, development and implementation the approach implies an understanding of the subjectively created world in the form of an ongoing process. Both the general form of phenomenology, i.e. ‘philosophical examination of the foundation of experience and action’ and hermeneutics, i.e. ‘interpretation and understanding of the context of our social environment in a manner akin to our interpretation and understanding of text’ (Winograd and Flores, 1990, pp.9 and 27-8 respectively) have philosophical commitment to anti-positivism. Equally, they all operate within the ‘interpretive’ paradigm for social inquiry (or organisational investigation) and analysis. In attempt to minimise risks in the development and implementation of ISs in an organisation Action Research (AR) strategy is suggested here as a means to enable the ISs researcher (or analyst/developer) to be implicitly and actively involved with the relevant group in the subject of investigation. Comprehensive details about AR are available elsewhere (see: Rapoport 1970; Foster, 1972; Susman and
The original concept of AR is credited to Lewin (1946), who expresses concern that the traditional science approach to social inquiry was not helping to resolve critical social problems (Susman and Evered 1978, p.587).

4 POSSIBLE USE OF ACTION RESEARCH FOR MINIMISING RISKS IN ISs

Figure 1 represents an AR framework which may be employed in the process of minimising risks in ISs. The various stages (1 - 6) represent the life-cycle of an IS. The model recognises that: (a) organisations are not homogeneous but they are different and unique in many ways; (b) clients (or managers) of an organisation may not fully know what they want (in terms of ISs) for their businesses; (c) the assumption should not be made that all managers in organisations are capable of articulating their expertise. Therefore, the AR model is based on an iterative learning process and implicit participation in problem identification/solving sessions between the organisation participants and the IS analyst/developer in order to have a better understanding of the domain, effect change and reduce (or avoid) risks in an IS project. As shown in Figure 1, the AR stage (No. 2) has an iterative link with the stage above it (No. 1) and the majority of the stages below it (Nos. 3, 4 and 5). With stage 6 inclusive the model is a single coherent ‘whole’ aimed at reducing (or entirely avoiding) perceived risks in the various stages of an IS development project. In most cases the effective involvement of the organisational participants from the initial stage to the final stage may result in little or no further serious training programme. If the approach is properly employed a minimum level of risks coupled with satisfactory performance may be expected in an IS project. Potential benefits of the application of AR are now discussed in the sections hereafter.

5 POSSIBLE BENEFITS OF ACTION RESEARCH STRATEGY

In most situations of organisational management data required from managers (or business ‘owners’) which are associated with types of risk in ISs and business risk factors are often a mixture of quantitative and qualitative parameters (see section 2). The assumption that the business ‘owners’ (or clients) fully know what they want and can clearly articulate their needs or expertise may be an over ambitious expectation. The benefits derived from using AR may help to obtain suitable data as well as minimise risks in an IS development and implementation processes. Some of the possible benefits of AR are enumerated below, substantiated with the work of other professionals and commentators.

5.1 Bringing about change

Experience in using AR shows that it has the potential to assist in identifying key elements of risks, business needs and data considered suitable for the development of an information system in an attempt to improve and simplify business decision making and operations. This involves investigation into what the managers may consider to be the main components of
Minimising risks in satisfying organisational needs

risks associated with profitably satisfying their customers' requirements as well as how to carry out such risk assessment/analysis in practice. The action methodology of research as a framework for inquiry in human activities has been suggested by various authors and practitioners (e.g. Lewin, 1946; Rapoport, 1970; Foster, 1972; Susman and Evered, 1978; Checkland, 1981; Checkland and Scholes, 1990). This knowledge about AR and personal experience of using it in the current IS project indicate that AR might be a suitable approach that may help to bring about change and minimise risks in organisational ISs, if properly employed. Fundamentally, AR does not view human actors as objects of inquiry but as initiators of actions in their own right that can bring about changes (Checkland, 1989, pp.38-9).

Figure 1 Possible Model for Applying AR to Minimise Risks in ISs Life Cycle.

5.2 Collaborative Learning in Risk Assessment

Action research philosophy proposes an iterative process of investigation and involves participative learning between the researcher (or analyst) and client (see: Lewin, 1946; Rapoport, 1970; Foster, 1972; Susman and Evered, 1978; Stowell, 1991). The concept is in concord with the notion of Soft Systems Methodology which articulates a process of inquiry that leads to action (Checkland, 1981; 1989; Checkland and Scholes, 1990). The integrative problem solving approach is considered a suitable means of reducing risks in IS projects.
such that: (a) the 'owner' of the problem situation and the information systems researcher (or analyst/developer) can collaboratively work out the nature of the problem(s) in a project of an IS development, its implementation, and the process of resolving the entire problem situation; (b) the organisational participants and the researcher can be involved in the process of learning and improving the system under investigation at an early stage of the project, thereby creating a feeling of ownership and satisfaction for the clients and analyst(s). The process of iterative learning through understanding, interpretation and experience of the problem situation by both the analyst and the organisational participants is comparable to the idea of hermeneutics, phenomenology (Winograd and Flores, 1990; Burrell and Morgan, 1994) and Vickers' concept of appreciation (Vickers, 1965).

5.3 Integrating Theory and Practice

The main concept of AR is that of combining theory with practice as the researcher acts on the social system. This has been shown to involve a cyclic process having five major stages: diagnosis, action planning, action taking, evaluating and specifying learning (see: Susman and Evered, 1978, pp.586-9; Stowell and West, 1994, p.128). The merging of theory and practice then subsequent reflection leads to an increased understanding of the problem situation, which may lead to appropriate action. The AR approach falls into the 'interpretive paradigm' as opposed to the 'functionalist' (positivist) paradigm of resolving organisational problem situations and is capable of assisting in reducing (or entirely avoiding) risks in the development and implementation stages of organisational ISs.

5.4 Creating a More Desirable Information Systems

In discussing the corrective measures offered by AR Susman and Evered (1978) note that 'the consequences of selected actions cannot be fully known ahead of time' (ibid p.590). This implies that in the development of an information system for an organisation, the researcher (or analyst/developer) should recognise that what the suitable system should be and how it should be developed to meet the client's needs must be deduced from the AR process itself and not assumed. An assumed 'what' and 'how' in the development of an information system is likely to lead to the creation of an unsatisfactory system which could even assault the very situation it is meant to improve or save.

6 THE APPLICATION OF AR AND RESULTS OBTAINED

Due to the complex nature of most management risk assessment activities, the AR-based method adopted in providing a framework for the organisational investigation and for Knowledge Elicitation (KE) is the Appreciative Inquiry Method (AIM). The process of KE is a difficult but crucial aspect in the development of KBSs due to the inherent problems of KE. Feigenbaum (1984) specifies the process as a critical bottleneck problem stage. Irrespective of the sophistication and suitability of the inference mechanism employed at the construction stage of the computer-based model of the risk assessment, the resulting Risk Assessment Decision Support System (RADSS) may introduce business risks; for example, if
the integration of the IT is not appropriately matched with the requirements of the organisation. Due to confidentiality, data considered sensitive to the organisation are excluded, but this does not diminish the value of the technique or the example presented below.

6.1 Eliciting Knowledge to Model the Risk Assessment Process

AIM provides a method of inquiry that enables both the information systems analyst (or researcher) and the domain expert(s) or client(s) to 'learn', identify and define their problem domain. The method enables the participants to pay particular attention to the use of data/information within the domain before attempting to transfer the data based on the expert's knowledge to a computer-based model. It is not the intention here to discuss the mechanics of AIM as they have been fully covered elsewhere (see: Stowell, West and Fluck, 1991; West, 1995) but to: (i) show how its efficiency, flexibility and practical tools may complement or replace the traditional interview approach in KE; (ii) show its application in the current project and the results obtained. The method draws its epistemology and practical tools from SSM through Checkland and Vickers' notion of 'appreciation' coupled with the process of 'appreciative' system (see West, 1995, p.140). The method is made up of three major phases that correspond to three main activities involved, from which a meeting at each of these phases, is arranged between the information system analyst/developer and the domain expert.

Applying Phase 1 to Investigate the Risk Assessment Domain

At the first session, the researcher (or analyst/developer) showed an example of a 'systems map' to a respondent as a possible procedure to present his/her view on paper. Based on a concise statement Risk Assessment within a 'central bubble' (such as in Figure 2) as a starting point, the client was asked to represent his/her view of the domain of investigation pictorially in the form of a 'systems map'. The main risk factors were represented by each respondent as 'primary bubbles' and their environmental influences were added as associated bubbles similar to those shown in Figure 2. The method helped the domain experts to express their own views of the domain of risk assessment without interference from the analyst or knowledge elicitor. At the same time it enabled the analyst to facilitate the inquiry process and support the client with a framework which allows the client freedom to express and represent his/her view within the boundary constraint of the 'systems map' itself. The elements of the 'map' drawn by the expert form the basis of a discussion. The results obtained from this phase was a map that gives a full but relatively low level insight into the expert's thoughts about the defined situation of focus in the original central element (or bubble). Away from the inquiry session, the analyst carried out a careful examination of the various maps produced by the different respondents in a particular department (or section) and developed a composite systems map (CSM) of the domain. An example based on a resulting CSM from investigation in one of the departments of the organisation is shown as Figure 2. The CSM encapsulates the elements from the clients' individual systems maps, given as: price, reliability, service and support, performance, technology.
Applying Phase 2 to Investigate the Risk Assessment Domain

The aim of the second phase of the organisational inquiry was to further explore the data/information given in Phase 1. The individual elements of the domain expert’s map were grouped or looked upon as single entities and each was redefined as a purposeful human activity (or entity). A detailed activity description was carried out based on the SSM mnemonic CATWOE as a guide for further inquiry (see detail in: Checkland 1981, Checkland and Scholes, 1990). Away from the venue of the inquiry and through the process of applying the technique of SSM, each of the purposeful human activities was then explicitly described by the elicitor, a description usually referred to as Root Definition (RD). Such a RD was produced for all the primary elements. The next step in this phase, which was also carried out away from the domain of inquiry was to convert each RD into an activity diagram, referred to as conceptual model (CM), using the conventions in SSM for the creation of the conceptual models. The CM was represented in a format suitable for return to the expert for the purpose of validation, deeper exploration of the domain and further discussion on the elements so far identified. A conceptual model built from the RD of the purposeful human activity (of risk assessment) to determine a product price through management practice as identified from the composite ‘systems map’ of Figure 2 is given as Figure 3. In the project, such a CM was developed for the other primary activities (or entities) of the Figure 2. Each CM serves as an agenda and a means for further exploration of the domain of investigation at the third stage.

![Figure 2](image-url)  
Figure 2: Example of a Composite Systems Map Based on the Results Obtained.
Applying Phase 3 to Investigate the Risk Assessment Domain
At the third stage the analyst took the conceptual model(s) back to the domain expert(s) for evaluation. The CMs helped the analyst to structure relevant questions about the situation under investigation and to form an agenda for further discussion. For an example of how to derive questions for discussion see (Stowell et al, 1991, p.162; West, 1995, p.155). A combination of the CMs, further discussions and explorations between the analyst and domain expert(s) helped to provide a better understanding of the domain of investigation and a platform for further data extraction. The CM has the advantage of providing a forum for the client to reflect on his/her view, opinion and data/information already given about the domain. Equally it was possible to correct any imperfect understanding of the risk assessment domain on the part of the elicitor/analyst through the discussions.

6.2 Further Work

Data/information obtained through the method have so far proved encouraging in their suitability for developing a knowledge-base for a customised model of an IT-based RADSS. The resulting prototype will be made available to the organisation for evaluation and necessary feedback. There has been a debate about difficulties and loss of data in linking CMs to Data Flow Diagrams (DFDs) among various authors (e.g. Doyle and Wood, 1991; Prior, 1991). The technique of DFDs was employed as a means to further cross check the possible linking of the data obtained and CMs to technological design and development. Details about DFDs are available elsewhere (e.g. Sawyer, 1991; Mingers, 1995; Gane and Sarson, 1979).
7 CONCLUSION

The findings of the investigation presented in the paper indicate that risks associated with organisational ISs can in turn affect other business activities in both product and service companies. Apart from the frustration this may cause to managers and other staff of the organisation there could be potential economic pitfalls in both the short and long terms. In some cases the commercial stability of the organisation may slide on a downward slope due to performance failures of the ISs which may in turn lead to inadequate customer satisfaction, hence the company’s poor competitive edge in the market-place. The point has been made that due to human ‘cultural differences’ organisations are not homogeneous, but are different and unique in their own respect including the way that business activities are carried out. The major issues advanced in this work are: (a) that IT has the capability to improve ISs if IT competencies and IS methodologies are appropriately combined; but a risk assessment/analysis model focused mainly on the feasibility stage of an IS project rather than on the ‘coherent whole’ may not adequately minimise overall risks for an organisation; (b) that AR strategy which considers a problem situation and articulates a learning process for both ISs analysts/developers and clients has the potential to reduce risks associated with ISs for an organisation. It is worth noting that while computers may be good at processing data, only human beings make things ‘happen’ in an organisational environment and suitably merging human capability and IT potential may help to reduce risks in IS projects. The approach presented is capable of coping with both the ‘soft’ (or sociological) area of human activities and the ‘hard’ (or technological) aspect of an information infrastructure system project, in order to minimise (or avoid) organisational risks.

ACKNOWLEDGEMENTS

Appreciation and gratitude are offered to Alex Paterson of Renfrewshire Enterprise, Paisley and to the Manufacturing Director, Ian McNair, and other management personnel at Compaq Manufacturing Limited, Bishopton, for their support in the ongoing project.

REFERENCES


Simulation supported business process performance assessment

Rajala, M. M.
Neste Oy
P.O.Box 20
FIN-02151 Espoo
phone +358-20 450 4725, fax +358-20 450 4352
Email Mikko.Rajala@neste.com

Abstract
Information integration, process integration or even the synergy between these two do not have their own intrinsic value. How they are experienced by the customer and how they serve the company adds to their significance. When building an integration framework or when rethinking business processes the value to the customers should be the guiding factor and the performance should be assessed from this viewpoint. In practice this is however difficult with the current modelling approaches being rather more structure than value oriented.

This paper presents a new way to combine existing familiar methods from different disciplines to achieve the customer orientation in rethinking business processes. The combination of methods also helps to identify the shared set of parameters in the business processes to respond to changes in an effective, timely and process-wide coherent manner. Combined methods support the cognitive process of human beings, where knowledge on enterprise, constraints of operating environment and goals of modelling are synthesised to produce new design alternatives. On the other hand, the approach takes advantage of computer simulation showing the real world potential hidden in different design alternatives. The methods are tested in a real world case of InterNAPS Ltd. in the process of handling key document-type (the Documentary Credit) information in the network of delivery logistics.

Key words
Business process, design, integration, value analysis, assessment, variation, simulation.
1 INTRODUCTION

Attaining better business process performance raises many questions concerning the means for performance planning. In order to design and maintain superior performance it is substantive to know the constraints of an existing process and its operation environment. The company has to identify internal and external discontinuities, customer and stakeholder requirements for understanding the operation environment (Collins et al., 1991). It is essential to reconcile the outcome of current business processes to fit external requirements, market situation including competitors and internal capabilities. In practice this leads to a modelling task in which the behavioural properties of the current process and its future alternatives are studied (MacArthur et al., 1994).

Enterprise models are produced through a cognitive process (see Figure 1), where knowledge on enterprise, constraints of operating environment and goals of modelling are synthesised to produce new design alternatives (Smith and Browne, 1993). Due to diverse design goals different model representations may be used to provide aid for design decision making. In this paper a new strategy for business process performance assessment is introduced. The strategy is suited for a form of design approach, which uses value analysis (Miles, 1972) for design processes, failure mode and effects analysis (FMEA) to point out weaknesses of the current process (MIL-STD-1629A, 1980), and computer simulation to measure the operation of new design alternatives. The approach to value analysis was basically designed for product development but has since been extended to functional cost analysis (Yoshikawa et al., 1994) and business process re-engineering (Rajala and Savolainen, 1995). Equally, FMEA was originally developed for reliability engineering (Soin, 1992) nowadays being extended to failure mode analysis of business processes (Rajala, 1995).

The design strategy can be put in a five step procedure:

1) Identify the goals and constraints of enterprise task environment.
2) Develop model representations for the current business process.
3) Determine performance characteristics of the current process.
4) Generate model alternatives for the current process.
5) Evaluate different models through simulation supported value analysis and select solutions for further development and implementation.
Figure 1 Design process in a constrained environment.

The relevance of the new approach to business process design is shown through a case of InterNAPS Ltd, which is a joint venture company owned by Finnish Neste Corporation and Russian InterTechnology Ltd. producing solar panel components for solar energy systems. This paper shows, by using the listed design principles, how familiar methods for product planning can be used in designing business processes, and how computer simulation can be used in assessing different design alternatives in a real world case.

2 SIMULATION SUPPORTED BUSINESS PROCESS PERFORMANCE ASSESSMENT - A REAL WORLD CASE

2.1 Goals and constraints of enterprise task environment

Neste Advanced Power Systems (NAPS), engaged in the development and global marketing of solar and wind power systems, has production facilities and operations in many countries, for example Finland, France, Russia, and Thailand. InterNAPS Ltd., a joint venture operating in Russia, produces crystalline silicon photovoltaic components for solar panels. Components are either sold to third part companies or assembled into solar electricity panels by NAPS-Solartron Manufacturing Thailand Ltd (NSMT).

In Figure 2 an extract from InterNAPS enterprise model is presented. The business process called “Sales management” is described using hierarchical IDEF0- model (FIPSPUB 183, 1993). The “Sales management”-process of InterNAPS consists of six subprocesses: A1: “Make offers”, A2: “Agree on trade terms” and A3:”Enter order”, A4:”Confirm order”, “A5:”Make the Documentary Credit”, A5:”Transfer order for processing”. In the first subprocess an offer is made to a customer and in the second more precise trade terms are determined. In the third and fourth subprocesses the customer’s order is accepted and
confirmed, respectively, and in the fifth the terms of documentary credit and payment are determined. The sixth subprocess takes care of transferring the order for delivery logistics.

![Diagram of InterNAPS sales management process](attachment:image.png)

**Figure 2** Main functions of InterNAPS “Sales management” process.

The InterNAPS sales process is dependent on international agreements (ICC, 1993) concerning Documentary Credit (DC) trade, primarily ruled by banks and financial institutions. These agreements include many constraints to be taken into account. InterNAPS has to follow the agreements making the redesign work even more challenging. The business, based on use of Documentary Credit in practise, means that customer and supplier have to make contract according to certain formats. The banks (both customers and suppliers) are responsible for all details in the contract specified in the regulations. Customers redeem the products from customs of destination port using DC-documents. Special attention is paid to document names, written in accordance to the DC-document.

The document based trading causes many practical problems to the sales function of InterNAPS Ltd. These problems were processed by using documented Failure Modes and Effects Analysis worksheets (see Table 1). The basic problem in “Make the Documentary Credit”, as concluded from Table 1, is the duration of DC handling process performed by banks, InterNAPS and the customer. The length is mainly due to banks, but a remarkable amount of time and money is also wasted in correcting DC documents. This is partly caused by international DC regulations requiring that every detail in DC, for example document names or weights of packages, are written in exactly the same way as in corresponding shipping documents. An associated problem is the variation of time needed to perform the
DC handling. Both problems cause difficulties in scheduling of delivery logistics (see Table 1). The customer register difficulties as delayed deliveries. Therefore, ability to control the time for the DC-making is important.

The objective of redesign was to make “Sales management” more efficient in terms of time and money. It means keeping the order processing under better control by minimising the variation of processing time and costs, and to reduce the negative effects on manufacturing scheduling.

Table 1 An extract from Failure Modes and Effects analysis

<table>
<thead>
<tr>
<th>Failure mode</th>
<th>Cause</th>
<th>Effect</th>
<th>Effect Amount</th>
<th>Corrective action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handling process takes too long with the banks</td>
<td>Banks want to make money</td>
<td>Cash flow planning gets difficult, production get problems due to lack of cash</td>
<td>Time delay 1-3 weeks</td>
<td>Follow principle “Do it right at once”. Make clear guidelines for defining the terms of DC</td>
</tr>
<tr>
<td>There is a need for close correction with every DC</td>
<td>Documents not written correctly</td>
<td>Extra costs due to changes, allocation of delivery logistics is difficult</td>
<td>Costs (100- 1000 $)</td>
<td>Ready filled model created considering documentary credit</td>
</tr>
</tbody>
</table>

2.2 Performing Value Analysis (VA) to generate new designs for performance assessment

On the basis of analysed behaviour of process network (see Table 1) the decision was made to select a subprocess starting from “Enter order“ as an object for redesign studies. As recalled “Enter order” takes in the customer’s order. The order is passed on to “Confirm order”. “Make the Documentary Credit” takes care of the correct terms of Documentary Credit. Finally, “Transfer order for Processing” puts orders into a process ending with product shipment. The DC- management is, as noted above, a process in the extended enterprise environment containing time spending subprocesses performed by banks and the customer.

Determination of functional roles of process activities

The first step of VA is to define activities of the main functions and, as well, to define which the supporting functions are. From a process chain perspective it is obvious that “Enter order“ and “Make the documentary credit“ are the main activities most important for a successful deal with the customer (Table 2). “Confirm order“ and “Transfer order for processing“ are supporting activities. Focusing the redesign work essentially on “Make the Documentary Credit“ is thus suggested, because the “Enter Order“ is a rather straightforward subprocess and does not need any redesign effort.
Determination of the objective for process redesign via functional cost analysis
The suggestion to focus on "Make the Documentary Credit" also arises from the functional costs analysis performed parallel with average lead time analysis. According to Table 3, the process of Documentary Credit management ("Make the Documentary Credit") causes the biggest expenses also being a subprocess with a long lead time. The goal of process redesign is now to develop such a process where DCs, with corresponding paper management, are done in a quicker and less costly way.

Table 2 Functional analysis: process chain view.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Action</th>
<th>Object</th>
<th>Main function</th>
<th>Supporting function</th>
</tr>
</thead>
<tbody>
<tr>
<td>A3</td>
<td>Enter</td>
<td>order</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td>Confirm</td>
<td>order</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>A5</td>
<td>Make</td>
<td>the documentary credit</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>A6</td>
<td>Transfer</td>
<td>order for processing</td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

Table 3 Functional costs - analysis incorporated with average lead time analysis

<table>
<thead>
<tr>
<th>Hours spent in the activity (h)</th>
<th>Cost due to activity (Cost units)</th>
<th>Order management (% of costs)</th>
<th>Make the DC (% of costs)</th>
<th>Order management (% of time)</th>
<th>Make the DC (% of time)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>670</td>
<td>91.3</td>
<td>86.4</td>
<td></td>
<td></td>
<td>Agree on trade terms</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>2.7</td>
<td>4.5</td>
<td></td>
<td></td>
<td>Enter order</td>
</tr>
<tr>
<td>4</td>
<td>44</td>
<td>6.0</td>
<td>9.1</td>
<td></td>
<td></td>
<td>Confirm order</td>
</tr>
<tr>
<td>35</td>
<td>335</td>
<td>21.3</td>
<td>23.3</td>
<td></td>
<td></td>
<td>Negotiate on terms of DC</td>
</tr>
<tr>
<td>4</td>
<td>45</td>
<td>2.9</td>
<td>2.7</td>
<td></td>
<td></td>
<td>Study contents of DC initiative</td>
</tr>
<tr>
<td>25</td>
<td>260</td>
<td>16.5</td>
<td>16.7</td>
<td></td>
<td></td>
<td>Fix DC</td>
</tr>
<tr>
<td>25</td>
<td>272</td>
<td>17.3</td>
<td>16.7</td>
<td></td>
<td></td>
<td>Accept DC</td>
</tr>
<tr>
<td>28</td>
<td>291</td>
<td>18.5</td>
<td>18.7</td>
<td></td>
<td></td>
<td>Inspect DC received from bank</td>
</tr>
<tr>
<td>32</td>
<td>360</td>
<td>22.8</td>
<td>21.3</td>
<td></td>
<td></td>
<td>Compare DC with order confirmation</td>
</tr>
<tr>
<td>1</td>
<td>13</td>
<td>0.8</td>
<td>0.7</td>
<td></td>
<td></td>
<td>Permit delivery</td>
</tr>
</tbody>
</table>
Generating design alternatives for current business process

During an idea generation session (IDEGEN++, 1995), the design team generated ideas of which two were selected to solve the problems connected to current process.

The first idea is to develop standard sheets and a database for DC and order handling, and then introduce these to customers, manufacturing units and transportation units. The idea is supposed to prevent the errors due to writing and is expected to shorten the processing time due to failures. The effect of this idea on process topology is that no heavy checking and rewriting for DC-documents is needed. Final cross checking with order confirmation can be skipped because all the documents and forms are stored in the database.

The second idea is to end negotiations on DC terms with customer before taking banks in to the process. The idea is supposed to shorten the processing time spent in banks due to elimination of extra processing iterations. The effect on process topology is that heavy rewriting of DC can be eliminated.

2.3 Evaluation of different models through simulation

The selected design alternatives were evaluated using computer simulation via the ServiceModel-package of ProModel Corporation (ServiceModel, 1995). Design techniques were used in the simulation experiment to plan the simulation runs. In these runs a warm-up period was used before the actual simulation. In the model it was supposed that personnel was able to handle either one (1) or three (3) parallel processes at the same time. The frequency of arriving orders was distributed either according to normal distributions N(35 h, 7 h) or N(70 h, 9h) and the corresponding number of arriving orders was either 10 or 30.

The simulation results of new designs were then compared to performance of the current process. Each model was evaluated by calculating relative performance profile (Gale, 1994) in respect to the present process (Table 4).

It can be seen from Table 4 that saved average time of system performance is remarkable in both cases. Performance increases about 53 % for model design alternative one and 50 % for model design alternative two (refer to Table 4 and Figure 3). Correspondingly, costs due to time spent on processing decreases about 67 % in design case one and 54 % in design case two.
Table 4 Design ideas with evaluation results obtained through simulation

<table>
<thead>
<tr>
<th>Idea</th>
<th>Effect of idea on system performance if realised</th>
<th>Effect of idea on process topology if realised</th>
<th>Average timed effect on system performance (simulated)</th>
<th>Average cost effect on system (simulated)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Develop standard sheets and a data base for Documentary credit and order confirmation, introduce these to customers, manufacturing units and transportation units.</td>
<td>Prevent failures due to writing, shortens the processing time due to failures. Makes it easier to allocate delivery logistics.</td>
<td>No heavy checking and rewriting for DC-documents needed. Final cross checking with order confirmation can be skipped.</td>
<td>- 53±26%</td>
<td>- 67±9%</td>
</tr>
<tr>
<td>2. End negotiations on terms of Documentary credit with the customer before taking banks in to the handling process.</td>
<td>Shorten the processing time spent in banks. Eliminate at least one iteration in processing. Makes it easier to allocate delivery logistics.</td>
<td>No heavy rewriting of DC needed.</td>
<td>- 50±22%</td>
<td>- 54±16%</td>
</tr>
</tbody>
</table>

Figure 3 Differences in costs and processing time (%)
2.4 Suggested new solution and its consequences to delivery logistics

As a consequence to suggested changes in “Sales management”- process, the lead time of order processing was supposed to be reduced significantly (50-53 %). The simulation results also indicate that the costs due DC-management would decrease effectively (54-67 %). Furthermore, the variation of costs, total time and working time spent in order processing was supposed to be lowered.

The above mentioned effects were obtained after implementation of the design alternative one, but not occurring in full scale as indicated by simulation. However, an additional but not minor benefit was perceived, namely due to smaller variation in lead times of order processing and DC-management the scheduling of manufacturing and logistics was easier to perform. The main result from the redesign and assessment effort was the ability of InterNAPS to more timely deliveries giving more value for customers.

3 CONCLUSIONS

A performance assessment procedure for business process designs is presented and applied to a real world case at InterNAPS Ltd. The procedure is based on co-use of human cognitive process supported by conceptual principles of design problem solving and computerised simulation techniques. When setting up the methodology, it was found that the human design process can be speeded up and design decision making confirmed by intellectual use of simulation. The foundation forms a ground of a strategy for computerised business process design. The strategy and the procedure are especially helpful in evaluating the differences between various versions of business process systems of extended enterprise. The other strengths of the proposed strategy are:

1) The approach connects human design processing with computerised evaluation of produced models.
2) Produced information can be effectively used in comparative performance analysis via modified VA.
3) Managerial decision quality is improved due to the consistency and objectivity of presented approach.
4) Simulation balances the inconsistencies caused by human judgement.

Process simulation through defined methodology arises important questions concerning methods to improve the efficiency of operations performed in the extended enterprise. The proposed design strategy reduces complexity of enterprise design problem by providing an analytical tool for decision making. With such tools more intelligent decisions on current and future enterprise systems can be made.
4 REFERENCES


FIPS Publication 183, (1993) Integration Definition for Function Modelling (IDEF0), Computer Systems Laboratory, National Institute of Standards and Technology, Gaithersburg, Md 20899, USA.


IDEGEN++ idea generation software and manual. CAC-Research Oy, Finland, P.O. Box 10, 02361 Espoo, 1990-1995.


5 BIOGRAPHY

Mr Mikko Rajala is currently working as a manager responsible for business process modelling at Neste Oil’s Information Systems Management in Espoo, Finland. Neste is a multidivisional company producing oil products, gas energy and solar power energy systems, and specialised chemicals for different applications. Mr Rajala has since 1987 acted as a research mathematician and since 1994 until 1996 as a enterprise modelling consultant for Neste’s business units. In his work he has also participated in different European working groups working on engine test methods to be used within CEC (Co-ordinating European Council for the Development of Performance Tests for Transportation Fuels, Lubricants and other Fluids). Mr Rajala received his MSc degree in 1985 and the Licentiate of philosophy (for PhD) degree in 1991 in Applied Mathematics from the University of Helsinki. Mr Rajala is currently finishing his studies for Doctoral degree in Industrial Economics at the Helsinki University of Technology.
Key driven business reengineering - How to get reengineering more efficient and effective

M. Stamminger
Information systems production planning, Volkswagen AG
Postbox 1832, D-38436 Wolfsburg, Germany
Tel.: +49/ 5361-9-21646,
Fax: +49/ 5361-9-21136,
e-mail: devw1bjk@ibmmail.com

Abstract
Hardly any management concept has been discussed as much as the reengineering of business processes and numerous articles using many of today's buzz words of business have been published. In spite of the wealth of information that is available many reengineering projects fail because the quantum leaps expected do not take place with conversion. The reasons for this are varied, but the common characteristic of failure is the lack of a systematic implementation approach.

This paper will present a methodical approach for the implementation of a key driven business reengineering program, which supports the planning, realization and controlling of reengineering measures. In this connection the role of an integral target system is described in order to identify the essential key factors and targets for an orientation of reengineering activities. Therefore, the choice of reengineering measures, whether the reengineering carried out should be evolutionary or radical, depends on the value added potential and the prioritization determined. Another aspect is the use of indicators as instruments for controlling in order to identify additionally needed reengineering measures. The result could be to reach new potentials and to get more benefits in reengineering, while decisions for investment in new resources, for example innovative information technology, will be more efficient and effective.

Keywords
Experiences and failures in reengineering, key driven business reengineering, business process reengineering, business process management, target system, key factor, indicator.
1 INTRODUCTION

1.1 "State of the art" and best practice in business reengineering

Shorter product life cycles combined with the globalization of the competitive environment require a permanent effort to increase productivity and a reduction of product development time (Time to Market). For most companies the pressure is becoming more intense to be faster and cheaper. This leads to a conflict of objectives for most existing organizations. To achieve more competitive advantages, many companies have focused their optimization activities on net product potentials (Bullinger, 1995). For this reason, more and more companies are defining new visions and are looking for economies of scale in their most important business processes.

When looking at the dynamic change of management concepts over the last decades, the transformation process in the companies can be split into different phases. Above all, the change in point of view is characteristic. Figure 1 shows the transformation over the years from rationalization to continuous improvement (for example Lean Management, TQM, etc.) to the renewal phase, including the concepts business reengineering and Concurrent/ Simultaneous Engineering.

While functional aspects were at the forefront of analysis and optimization in the past, emphasis has now shifted to a global view of business processes. Business processes are defined as processes, which have a strong influence on the fulfillment of customer requirements. In other words: “Everything the customer does not appreciate is a waste!”(Hammer, 1995). Therefore, the main objective is to achieve drastic improvements in business performance, which can be measured directly in the market. In this connection for a couple of years, the slogan reengineering draws attention. Hardly any management concept has been discussed in publications in the past as much as the term of business reengineering. Numerous concepts
Key driven business reengineering

with all sorts of term variations have been developed while companies are still looking in vain for the right approach, for solutions to all their problems (Homburg, 1996).

At the beginning of the 90's the economic crisis demonstrated that companies were not reaching the potential of change, necessary to achieve the stronger demands of the market (Bullinger, 1995). However, from practical experience, it can be determined that many reengineering projects fail or do not achieve the high level of success expected (Scott, 1994 and CSC-Index, 1994). A study in a hundred German companies showed that business reengineering has failed with its approach and that there is a general imbalance between expectation and reality (Homburg, 1996). Therefore, nearly all interviewed companies hoped for increased productivity, but only a quarter of them reached this goal. The essential reasons for failures can be described and categorized as following:

**Human aspects**

Hammer and Champy claim that many reengineering projects are threatened to fail because the management maintains classic leadership and organizational structures and do not rethink these fundamental principles (Womack, 1996). Studies in the USA illustrated that there is a causal relationship between the motivation of management and the achievement of the targets (Hall, 1993). Failure occurs, therefore, because the theory of a global view of business process is not thoroughly implemented. Functional barriers are still hindering the search for solutions linking all areas.

Another aspect causing failure is that persons affected within the processes are not sufficiently integrated into the reengineering activities. Thus many employees are unwilling to adopt this approach. They are uncertain and afraid about their job security due to the employee reductions made over the last decades. The identification and motivation for a common vision are lost by many employees. In this context Scott-Morgan (1994) argues that drastic measures with aggressive targets could not be enforced in most companies.

Further, in complex problems as we can identify them, the persons affected tend to disregard essential factors outside of their own organizations. This creates the risk that problems resulting from reengineering will not be observed and corrected because of this isolated view. The problems in the companies often are not focused through different perspectives, so they are not recognized sufficiently by the employees. The reason for this is that they are overloaded with daily problems and are not using their knowledge and experience to manage the situation (Dörner, 1989).

**Missing target orientation**

Very often the knowledge for completing a reengineering project is lacking. This hides the danger that in the analysis phase, i.e. the modeling of the actual state artificially complicated. is overtly extended thus pushing the real goal of the reengineering the definition and implementation of concrete solutions into the background. From practical experience it can be determined that often a poor systematic approach is found in the companies. Additionally a clear target orientation is not present. Furthermore, a continuous instrument for measurement and controlling of reengineering activities such as process analysis or definition, etc. do not exist. Thus it is almost impossible to check the efficiency of reengineering measures in order to minimize the risk of failure (Stamminger, 1996).
In practice concrete targets which are used as a basis for assumption in order to work out the right priorities for a target-oriented reengineering are often lacking. Quantum leaps expected do not take place because targets were set too low and market-oriented success parameters were neglected (CSC-Index, 1994). Often, however, the success parameters are not known which are necessary to achieve a breakthrough (Bullinger, 1995). In the search for these parameters or essential key factors for reengineering, the complexity of problems in the companies are often estimated incorrectly. The analysis of different perspectives and influencing factors between the processes is normally not taken into account. With regards to the global or local aspects in business processes, the possible benefits of reengineering measures could not be estimated through a single (one-sided) approach (Gomez, 1995). This hides the risk of sub-optimization, which means that the whole performance of the company could be disturbed. For example, when reengineering cost or profit centers, this becomes recognizable because competing targets can often be found which may negatively influence the strategic targets of the company. Besides, the mistake of structuring targets in form of hierarchical structures is still being made. This may be a main reason why contributing to erroneous conclusions regarding the prioritization of reengineering measures (Stamminger, 1996).

Problems in process modeling

Process modeling is a creative and complex task. The main objective is to model information about a defined part of the system which is to be considered. To achieve this, systematic methods and modeling techniques are required. In this respect, a whole range of modeling techniques (ARIS, IEM, CIMOSA, etc.) and computer assisted tools (ARIS, MOOGO, etc.) are now available to deal with the problems involved. However, from practical experience, it is often not clear whether and on which level the affected processes shall be modeled to get evident facts about organizational weak points, etc. Predominantly process models are being used for visualization and as a basis to discuss possible solutions through the team involved in the reengineering process, which the author will not criticize here. It should rather emphasize the danger that with only illustrating the actual state through, for example modeling information flows as well as marking weak points, etc. the reengineering work has not been completed. A continuous conversion of reengineering projects from analysis to the process definition often is not guaranteed and goes too lazy.

1.2 Objective of this paper

As discussed above it becomes obvious how many different aspects must be considered, to make business reengineering successful. Above all, the human resources must be better integrated in the cycle of reengineering. Consequently, there is an urgent call to action to set in motion changes in mind sets necessary to develop and implement common visions. In our companies this can only be achieved through a responsible action and personal conviction (Gomez, 1995 and Cooper, 1994).

Summarizing this, besides the human aspects the author sees the main potential in the procedure of reengineering to increase efficiency and effectiveness. In order to minimize the risks of failure in reengineering projects it is important to solve the problems in a company's organization in a systematic way. The following thesis will help to develop such a procedure:
Key driven business reengineering

2 KEY DRIVEN BUSINESS REENGINEERING

As discussed in the previous chapter there are many reasons why reengineering fails. To achieve more success a systematic approach is required. This article presents a methodical approach for a continuous planning, realization and controlling of reengineering activities. A target system forms the central core of this approach, which controls the reengineering through key factors and concrete targets. Furthermore, the procedure supports the controlling of reengineering activities and the definition of additional measures if necessary. The result is to carry out new potentials and to get more benefits from reengineering, while analysis, definition and implementation of the process itself is more efficient and effective.

On the basis of the thesis listed above, the global cycle of business reengineering is introduced in a first step. As shown in figure 2 the cycle is subdivided in two areas Business Process Reengineering and Business Process Management. While process reengineering includes all activities which have to do with the modeling of business processes, process Management is understood as controlling the processes on the operative level in a company. This means that process Management takes on the function of an early warning system, in order to obtain information about actual events or trends for future anticipation (Krystek, 1993).

As mentioned above the target system plays a significant role in the cycle of business reengineering and serves for orientation and controlling of reengineering measures. Through its results, the target system forms the base to determine whether and where reengineering measures are necessary. Furthermore, it determines whether the reengineering carried out should be evolutionary or radical. Therefore it supports the various phases of business reengineering from:

- process analysis,
- process definition via
- implementation to
- continuous improvement entirely.

It is characteristic for the circle displayed in figure 3 that the several phases cannot be seen purely as a sequential series of activities which are passed through uniquely, but also as a networking problem solution process with iterating action steps. The results within the phases can be evaluated by a constant comparison of the actual state with the target set. This allows
for interaction at anytime in the reengineering process with additional measures if the deviation is determined to be too large. Through the continuous interaction of the target system the cycle of business reengineering is considerably more insensitive to changes of basic conditions or new impulses. Therefore, the danger of the whole project failing can be minimized.

To achieve more potential and efficiency in reengineering the essential success factors have to be found in the company's organization. Another important aspect is to transform these key factors into concrete targets. The definition of targets for a company is the basic precondition to identify facts as problems and to assess the effectiveness of solutions (Scholz, 1993). As shown in the previous chapter, in practice many deficits occur while specifying key factors and targets needed for a key driven reengineering.

![Figure 2 The complete cycle of business reengineering (Stamming, 1996).](image)

The following questions have to implement successful reengineering:

- How can the essential key factors be identified?
- Which processes shall be improved and which organizational units are affected?
- What effects on other processes can be expected through reengineering?
- What benefits are available and how high should the targets be set?
- Which indicators are needed to assess the performance of the processes and the quality of their outcomes?
- How can the results of reengineering measures be controlled?
The cycle of business reengineering starts by recognition of needed reengineering measures. This can be initiated through a culture change in the company, for example from a purely functional thinking to a process orientation. Another impulse initiating change can occur, if the indicators in process controlling determine that the performance goes out of tolerance drastically. In such cases measures within the phase of continuous improvement are not sufficient to solve these problems, thus the conditions in the company have to be checked. This is guaranteed through the several steps in the target (system) definition process as shown in figure 3.

Figure 3 Example for interaction between target system and business process reengineering.

For a better comprehension of the interaction between the target system and the respective phases of business reengineering the several steps of the procedure are described in more detail as follows:

System Thinking (1)

Today many companies are characterized through complex processes and structures in their organization. Additionally they actually occur with many combinations and are influenced or disturbed dynamically through many factors. These kind of systems are called complex systems, which have the property that they cannot be determined exactly (Gomez, 1995). A good example for such complex situation is the change of the markets from a vendor to a customer-oriented market. The fact is that the companies are now no longer independent and isolated from the developments beyond their environment (Krystek, 1993).
From practical experience it can be determined that the analysis of complex systems is a difficult task because we tend to neglect important aspects while searching for solutions quickly without reaching an integral position of the problem and aspects belonging to it (Gomez, 1995). One example of this is the structuring of success factors in the form of target hierarchies. Often targets are developed from strategic aims of management through a top-down approach. This hides the danger that real relations of cause and effect of complex situations, especially of business processes, will not be sufficiently realized.

The method of "System Thinking" supports the analysis of the most important influences and relations in complex situations. Furthermore, it is possible to visualize these aspects and to check the intense and dynamic influence of the system components (Gomez, 1995). To achieve this relevant aspects of process actions should be considered from different perspectives within the company. As a consequence, it is necessary to determine suitable and essential key factors, which allow a balance between global and local relations in the company. Afterwards, these key factors including their relationships to one another can be visualized with cause and effect diagrams.

Interpretation (2)
After the presentation of the key factors in the cause and effect diagrams it is necessary to analyze the relationships through a qualitative and the quantitative valuation process. To achieve the correct success parameters, and thus priorities for reengineering measures, the intensity of the relations among the key factors has to be analyzed. Depending on the influence to and from other factors an active and passive value can be evaluated and presented in a special diagram. This distinction is of considerable interest, because the separate values are then classified so that the critical factors can be identified to establish priorities for a key driven reengineering. It is obvious that it makes only little sense to concentrate the activities on factors which have only low influences on others, or where an interference itself is very difficult. To summarize this, key factors are defined as parameters which have a large influence on others. These are most suitable for the introduction of reengineering measures because they possess a high potential for improvement and build on the assumption for successful business reengineering.

Identification of key processes (3)
If the key factors and the priorities for reengineering measures were specified it is obvious that for each key factor the processes affected have to be identified. This simply means which processes in the company have a major influence on the key factors through their outputs and process performance. For example, if the "start of production" was identified as a critical factor, the subprocesses of purchasing of manufacturing equipment and/or engineering change management could be affected. Another aspect to be taken into consideration is the definition of suitable indicators. Indicators make measurements possible so they are used as controlling instrument because they help to register deviations from the targets set (Fries, 1994).

Process analysis (4)
In a next step, depending on the results of the determined success parameters, specific aspects of the processes and organizational units affected can be analyzed in more detail with the
methods of process modeling. Concerning the analysis of processes, the flow of information is at the center of observations. At this point it is interesting, which information is exchanged when and how, in which form and how often, by whom and for whom it is prepared etc.. Furthermore, it is required to model the information flow in an entire and logical context. To achieve more transparency in process action and to get more information with regard to weak points it is necessary to measure the performance of the processes on the basis of the defined indicators. This means that all input and output relations regarding cost, duration (including the quality of outputs) have to be evaluated. This is the basis for a better estimation of reengineering potentials and target definition.

Target definition (5)
To get an orientation for reengineering it is important to check the results of the process analysis against improvement potentials in order to set priorities. Based on the established potentials and actual performance of the processes affected, in the next step targets must be concretely defined. Therefore, the choice of reengineering measures, whether the reengineering carried out should be evolutionary or radical, depends on the value added potential and the prioritization set through the targets determined. The ability of quantifying the targets makes it possible to exactly measure and evaluate the important facts. The targets are specified through a value or defined value range. To better utilize the target definition process the benchmarking concept is helpful. It has been introduced successfully in the past and allows the comparison of facts between other companies, especially competitors. Thus it is possible to recognize trends or changes earlier.

Evaluation (6)
As described already, the cycle of business reengineering is characterized through several evaluation steps after each phase. Thus the iteration within the procedure is supported by the results of reengineering activities itself. For example, the progress within the phase of process analysis can be checked continuously if the specified processes were analyzed completely. This is important if the conditions in the reengineering project change so that new requirements and aims from the target system can occur. For example, regarding additional activities in process analysis. Therefore it is necessary to be able to interfere at any time of reengineering in order to set new priorities. The continuous interaction of the target system with the respective reengineering measures has the effect that the procedure is more insensitive to interference and/or new impulses from outside. Thus the evaluation steps make it possible to minimize failures and maximize the contributions of sources for a success reengineering sources.

3 CONCLUSION

To sum up one can see that business reengineering can be controlled through a systematic and key driven approach. A further characteristic of this approach is the integration of Business Process Reengineering and Business Process Management. Through the cycle of business reengineering results of the several phases can be evaluated at any time. Thus decisions
concerning analysis and definition are oriented entirely to essential key factors. The choice of reengineering measures, i.e. whether the reengineering carried out should be evolutionary or radical, depends on the value added potential and the prioritization determined through the targets. In process performance measurements, indicators play an important role in order to make efficiency and effectiveness of business processes more transparent and serve for a better conduit of information for all those involved in the processes.

4 REFERENCES


BIOGRAPHY

Dipl.-Ing. Markus Stamminger, born in 1966, studied mechanical engineering at the Technical University in Karlsruhe, Germany. Main subjects were production engineering and integrated CAD/CAM-solutions. He joined the Volkswagen Group in Wolfsburg in 1994. In the project IMPULS he is involved in reengineering the process of production preparation for new vehicles. Based on his practical experiences he writes his dissertation about systematic approaches for business reengineering in the automotive industry.
PART FIVE

Information Infrastructure and Engineering Processes
Generation of milling data in a virtual manufacturing framework

M. Matsuda
Faculty of Management and Informatics, Sanno Institute of Management
1573 Kamikasuya, Isehara, Kanagawa 259-11, Japan,
Telephone:81-463-92-2211, Fax:81-463-93-0554,
E-mail:matsuda@mi.sanno.ac.jp

F. Kimura
Department of Precision Machinery Engineering, Graduate School of Engineering, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113, Japan,
Telephone:81-3-3812-2111, Fax:81-3-3812-8849,
E-mail: kimura@cim.pe.u-tokyo.ac.jp

Abstract
The framework and method of implementation for the automatic generation of optimized milling data with guaranteed accuracy are discussed in this paper. By analyzing the milling data generation process, the transformation of the product description has become transparent. Namely, milling data is generated based on machining features which implicitly include a product description at the product design stage using design features. A trial milling data generation system was implemented, which can directly receive the product model with design features, convert it automatically into the required description of machining features, and produce machining data based on machining features. This trial showed that the process of milling data generation based on product description data is the same as doing the milling on the computer. Then the structure of a virtual milling system mechanism is proposed by the conceptual expansion of the implemented milling data generation system. This proposed system would generate milling data of high quality, using the product model and the milling machine model.

Keywords
product model, milling machine model, virtual manufacturing, process/operation planning, machining preparation, NC data generation
1 INTRODUCTION

Recently, it has been said that virtual manufacturing will lead to the next generation of manufacturing systems. However, the development of new technologies are required in order to implement the entire virtual manufacturing system. Virtual manufacturing is not a simple simulation system. It is used as a tool for generation, transformation and confirmation of models or data which are required for production. In a virtual manufacturing system, various types of models are used and processed: for example the product model, physical process model and activity model. Methods for organization and processing of these models are key technologies to realize the virtual manufacturing system (Kimura, 1993a) (Kimura, 1993b). When this manufacturing concept is applied to machining preparation, the level and quality of automation in machining would be improved. That is to say, if the machining data generation system could guarantee machining results using its output data, the trial process could be omitted. This means that a large variety, small volume production, or even single product production, which do not desire machining trials, would be realized.

This paper provides the basis to the future machining system which generates machining data with guaranteed accuracy, by considering concretely a milling data generation system based on a virtual manufacturing concept. By focusing on the transformation of the product description in a milling generation mechanism on a computer, the structure of the milling data generation system becomes transparent.

2 FLOW FOR MACHINING DATA GENERATION

2.1 Traditional flow

In the traditional flow, human operators of the machining data generation system must subsequently translate design drawings into terms acceptable by computers, and input this information to get the machining data. In other words, a human operator selects the tool, extracts the corresponding shape to be machined by the tool, and then translates the product description into the desired format.

If the product designer uses a CAD (Computer Aided Design) system, human operators are still required to input some data used by the machining data generation systems including CAM (Computer Aided Manufacturing) systems. As shown in Figure 1, usually the final shape data of a product which is output from a CAD system automatically becomes the input to a CAM system. However, a CAM system requires not only shape data but also designations for the machining area, data for tools used and so on. The latter information are input by human operators who think about the facility of the factory and derive these data.

2.2 Transformation of product description

When a CAD system is used, the product model created at the design stage should be given as input data in the machining data generation system. The product designer usually describes the product using design features. However, most machining data generation systems accept only machining features in product descriptions. The automation of machining requires that the
construction of a product model focuses on machining features. Here, the term “machining feature” means a volume machined by a single tool, such as a drilled hole or milled area. On the other hand, “design feature” means a unit to achieve some function, such as transmission of motion, transformation of motion and parts fixturing.

A product description created at the product design stage using design features implicitly includes machining features. However, explicit descriptions of machining features are required. In this case, the process of machining feature conversion/extraction becomes necessary. The conversion/extraction process derives explicit descriptions of machining features from descriptions of design features. Machining data which control machines and produce actual products, are generated based on these machining features. These transformations of product descriptions are shown in Figure 2.

![Diagram](image_url)

**Figure 1** Traditional flow for machining data generation.

**Figure 2** Transformation of product description.
3 MACHINING DATA GENERATION BASED ON PRODUCT MODEL

The determination of machining features from a product model is straightforward if the form features created by a designer match the machining features recognizable by a process planner or operation planner. However, if the design features do not match the recognizable machining features, the product data include information on machining features but not explicit descriptions of them. If the machining data generation system could directly receive descriptions of the model, and convert them automatically into required descriptions, a total system of machining data generation would be possible.

Figure 3 shows the conceptual structure of the above mentioned system for machining data generation. The product model which has a description of design features is the input to this system. The system sorts the design features into two groups. One group consists of design features matching with machining features, such as a hole being produced by a drilling machine. Design features in this group are called drilled features. Drilling data is generated based on these drilled features. The other group consists of design features not matching with machining features, such as a pocket which is produced by a milling machine. Design features in this group are called milled features. A milled feature consists of free-surfaces and planes. A free-surface part is machined by ball endmill, and the part which consists of planes only is machined by a straight endmill. The system classifies composing elements of milled feature depending on surface type, and then extracts machining features by means of milled volume by one tool. Milling data is generated based on these machining features. Finally, the system merges these drilling data and milling data in order to get machining data.

![Figure 3 Conceptual structure of machining data generation system.](image-url)
4 EXAMPLE OF MACHINING DATA GENERATION SYSTEM

4.1 Structure of milling data generation system

A trial system for automatically generating milling data based on extracted milling features is developed (Matsuda, 1991). A milling feature is one type of machining feature. This system considers machining of a cavity for a die or mold. A cavity is one kind of pocket. This system treats only a pocket with a complicated 2 1/2 dimensional shape and with round edge, chamfer edge and tapered face. A 2 1/2 dimensional shape means the combination of several volumes which are derived by sweeping a 2 dimensional figure. This system produces a milling process plan from rough cutting to finish cutting, lays out the shape of a form tool for the finish cutting, and generates the corresponding NC data. In the rough cutting process, straight endmills are used. In order to realize high efficiency of cutting, tools with diameters as large as possible are used for rough cutting. At the intermediate cutting stage, bottom faces are finished using straight endmills. At the finish cutting stage, side faces are finished using form tools.

The structure of the milling data generation system is shown in Figure 4. This system requires form-feature descriptions of the cavity as given by product designers and information on available tools. The operation of this system proceeds through five steps. (1) The milling area generator determines cross sections at successive depths of the cavity's bottom faces. (2) The tool selector chooses tools. (3) The tool-center paths generator determines tool-center paths for each cross section and calculates milled areas. (4) The tool-center paths editor reorganizes the milled areas to derive milling features. (5) The milling process plan and NC data are generated based on extracted milling features. The methodologies for extraction of milling feature and milling data generation are explained as follows.

![Figure 4](image-url) Structure of milling data generation system (Matsuda, 1991).
4.2 Computation of milled cross section at each bottom face

A cavity is described by the shape of each bottom face, the depth of each bottom face, the inclusion relationships among bottom faces, and data for taper, round and chamfer. Bottom faces are represented using 2 dimensional B-reps solid models. The depth of each bottom face is described as an attribute of each bottom face. The inclusion relationships among bottom faces are described using two predicates: “include” and “duplicate.” In Figure 5, face2 is an island within face3. According to the top view of this figure, edge12 of face1 and edge30 of face3 are duplicated. The data for taper, round and chamfer are described as attributes of each edge.

The system processes a series of milled cross sections with the milling area at each bottom face depth. Computation among bottom face shapes to get a milled cross section depends on depth information of bottom faces and inclusion relationships among bottom faces. In Figure 5, the cavity consists of three bottom faces. At depth a, the milled cross section shows the union of face1 and face3. At depth b, the milled cross section equals face3. At depth c, the deepest depth, the milled cross section equals the difference between face2 and face3.

![Figure 5](https://example.com/image) Computation of milled cross section.
4.3 Tool selection

At this stage, each milled cross section is considered independently as a milling area. For one milling area, the selection of tool diameters of a straight endmill for rough cutting is provided depending on the corner size, distance between island areas, and the diameter of the biggest available tool. The tool diameter is selected by choosing the biggest available tool on the list with a diameter that does not exceed the distance between island areas and the corner size, and also considering the milling allowance for the next cutting stage. The final candidate list of applied tool diameters consists in an ordering of tools according to their diameters, with the biggest possible tool providing optimum milling efficiency.

4.4 Determination by tool center path

Figure 6 shows the tool center path determination process to get a milled area for the corresponding tool at the corresponding depth. Following a proper sequence from the list of candidate tool diameters, a bigger tool is applied for the considered milling area as follows. (1) Initial peripheral tool path center lines are generated for the cross section considered. (2) Interference points in the initial tool path center lines are determined. (3) Parts of the center line that do not interfere with other center lines are collected in order to create loops. Expansion of the tool path center line loop becomes a milled area for the corresponding tool at the corresponding depth. (4) Some tool path center line loops are rejected because the cutting areas are too small. (5) Secondary tool path center lines are generated for the remaining areas with no corresponding tool path center line loop. (6) In order to complete all loops, ends of the secondary tool center line paths are connected by generating straight lines. The system repeats the loop creation process.

![Figure 6 Determination by tool center path](image-url)
4.5 Editing of tool center paths

A single endmill cuts the volume which was obtained by sweeping milled areas for one tool at a certain cutting stage. Such a volume is a milling feature. Figure 7 shows the editing of all milled areas to derive milling features. Milled areas which are provided by tool center path determination are divided into groups according to endmill diameter. These groups of tools are arranged by size, largest to smallest. In a group milled areas are arranged by depth, shallow to deep.

The process of deriving milling features for rough cutting, with a single tool of one diameter, is as follows. (1) Duplicated milled areas at each depth in one group are eliminated. (2) Tool path center line loops of milled areas are divided by the taper, round or chamfer size of the original face. Here, groups of milled areas for rough cutting are determined. One group of milled areas equals a milling feature. The process plan and NC data are obtained from these groups of milled areas.
Milled areas in one group are determined to get milling features for intermediate cutting. If two milled areas have a duplicated part as seen from the top view, there is no bottom face on the duplicated part. In order to eliminate the duplicated part, the difference between the two milled areas is computed on the tool path center line loop.

To get groups of milled areas for finish cutting, the depth of each milled area for intermediate cutting corresponds to the depth of the edges of the original faces. If there is no equal correspondence, that milled area is eliminated. One group is divided into smaller groups by the original edge attributes of the milled area, such as "round." Form tool designs are determined depending on these small groups. One small group corresponds to a milling feature for finish cutting.

![Diagram of virtual machining](image)

**Figure 8** Concept of virtual machining (Matsuda, 1995).

5 MILLING DATA GENERATION USING VIRTUAL MACHINE

5.1 Conceptual expansion to virtual machining

In order to produce actual mechanical parts, it is necessary to generate an NC machine program based on the product model created at the design stage. The product designer usually describes the product using design features. An NC data generation system usually calculates the cutter paths based on such input data as machining features and tools. The product model created at the design stage should be provided as input data to the machining data generation system. The machining data generation system should extract machining features and plan operations such as the selection of tools, determination of cutter speeds, and calculation of cutter paths. In a machining data generation system, this means product data with design features is translated into product data with machining features. In the previous milling data generation system, it was shown that the machining data generation process is a translation process from product
description with design feature into an NC machine program that corresponds with the machining features. Explicit descriptions of machining features are derived by applying machining constraints, such as machining methods and required tools.

These translation processes are facilitated by applying constraints on machining factors such as machining methods, machine specifications and machine functions. For example, the size of the tool diameter was the primary constraint in the previous milling data generation system. This also shows that the level of optimization and accuracy of machining data is proportionate to the number of machining constraints imposed and the accuracy of these constraints. The development of a translation system involves the implementation of mechanisms that apply machining constraints to product data. In other words, to realize this system it is necessary to build a virtual manufacturing factory within a computer and carry out machining in this virtual factory as shown in Figure 8 (Matsuda, 1995).

![Figure 9 Virtual milling machine.](image)

### 5.2 Virtual milling machine

A virtual machine becomes the component machine which carries out manufacturing in a virtual factory. A virtual machine is a computer model which represents the specification, function and behavior of a physical machine. A virtual machine is the controlled object in the virtual factory.

Generally, a virtual machine is composed of two models: a component model and a process model. The component model is the description of the static elements of the machine, such as machine specifications, tools, jigs or fixtures, and the controller. The process model is the description of the actual operation by means of simulation, such as cutter location, cutting force, cutting accuracy and error sensing. Each model has several variable classes. Through the combination of these classes, a flexible virtual machine modeling can be realized (Matsuda, 1995).

Figure 9 shows the virtual milling machine as a result of modeling. In this figure, the structure of a virtual milling machine is simplified, such that its components are limited to the
Milling data in a virtual manufacturing framework

A tool model consists of the solid model of the tool shape, static character data and a cutting process model by means of dynamic data which express tool working condition. Total data for milling conditions is obtained using tool models.

![Figure 10](image)

**Figure 10** Structure of virtual milling system.

### 5.3 Structure of virtual milling system for machining data generation

Figure 10 proposes the structure of the virtual milling system for machining data generation. A product model is input into the system. The product model description is translated into physical machine control commands by means of usage of the virtual milling machine. A previously proposed milling machine model is used as a virtual milling machine in the system.

In the proposed system, a milled design feature is decomposed into surfaces and restructured, depending on the endmill type used. For a free-surface a ball endmill is applied. For a plane a straight endmill is used. Next, the system extracts milling features using information from the milling machine model, such as a list of available tools. Then the system calculates cutter paths, decides cutting conditions such as cutting speed, by reference to information from the milling machine model, and then generates milling data. Furthermore, the system optimizes milling data by evaluation of milling condition data which is provided by the working of the virtual milling machine based on output milling data.

### 6 CONCLUSIONS

In this paper, the milling data generation system using the product model and milling machine model is proposed. In this proposed system, the product description is transformed to milling data by applying the milling process in the computer. In other words, virtual milling is done
using a milling machine model as a virtual machine. Input to the virtual factory is the product model which is obtained at the product design stage, and the output from the virtual factory is the NC data for milling. This milling data is optimized and guarantees the result of milling when it is used, because it is produced by virtual milling. Expanding the concept of this proposed virtual milling system leads to a general structure of a machining data generation system.

7 REFERENCES


8 BIOGRAPHY

M. Matsuda is an associate professor in the Department of Computer Science of Sanno Institute of Management. She received a Dr. Eng. degree in precision machinery engineering from the University of Tokyo in 1989. She has been active in the field of CAD/CAM, process/operation planning, NC programming and numerical control. Her recent research interests include product modeling for manufacturing and virtual manufacturing. She is a member of Japan Society of Precision Engineering and Information Processing Society of Japan.

F. Kimura is a professor in the Department of Precision Machinery Engineering of the University of Tokyo. He received a Dr. Eng. Sci. degree in aeronautics from the University of Tokyo in 1974. He was a research associate at the Electrotechnical Laboratory of the Ministry of International Trade and Industry from 1974 to 1979. He then moved to the University of Tokyo, and was an associate professor from 1979 to 1987. He has been active in the field of solid modeling, free-form surface modeling and product modeling. His research interests now include the basic theory of CAD/CAM and CIM, concurrent engineering, engineering simulation and virtual manufacturing. He is a member of IFIP WG5.2 and 5.3, and a member of CIRP.
Framework Services for Design Data and Design Flow Management

Rene van Leuken and Alfred van der Hoeven
Delft University of Technology / DIMES-DDTC
Department of Electrical Engineering
Mekelweg 4, 2628 CD Delft, The Netherlands
rene@cas.et.tudelft.nl, alfred@cas.et.tudelft.nl
http://www.ddtc.dimes.tudelft.nl

Abstract

In this paper we present an overview of principles and mechanisms which support management in their task to control and track the engineers work and which guide engineers through the creative design stage. In today’s engineering environments engineers use design tools to create, modify and verify their designs. During this work the primary focus of an engineer is to successfully perform design tasks. Design descriptions and the tools that operate on them become more complex every day. Today design descriptions are usually constructed hierarchically, consist of multiple levels of abstraction (views) and have a status. The central theme in our approach is that the framework maintains information about the design descriptions and the design activities, so called meta data. This involves primarily information on the presence of cells, or design objects, their relationships, their version history, and the operations that have been performed on them. The information model we use is the OTO-D data model. It is particulary suited to visualize object type hierarchies.

The principles and mechanisms are implemented in a set of framework services which are part of the Nelsis Framework. The Nelsis CAD Framework User Services allow engineers to interact with the framework, for instance to invoke tools or to browse information stored by the framework. The services include a hierarchy browser, a version browser, a design flow browser, and more.

Keywords

CAD frameworks, design flow management, integration platform, data modeling, user interface.

1. INTRODUCTION

Design descriptions and the tools that operate on them become more complex every day. Today design descriptions are usually constructed hierarchically, consist of multiple levels of abstraction (views) and have a status. Quite often an engineer wants to retain multiple versions of a design description. To manage these design descriptions and the tools that operate on them, a CAD framework is needed.
According to the CAD Framework Initiative, "a CAD framework is a software infrastructure that provides a common operating environment for CAD tools. A framework should enable users to launch and manage tools; create, organize and manage data; graphically view the entire engineering process; and perform design management tasks such as configuration management and version management."

A framework that provides these facilities, is the Nelsis CAD Framework. The Nelsis CAD Framework is a flexible, light-weight framework that enables the engineering community to exploit high performance environments. The research and development of the Nelsis CAD Framework concentrates on the following key requirements:

- Openness
- Design data and design flow management services
- Configurability
- Performance
- Data Distribution
- Domain Independent
- User Friendliness

![Figure 1. The graphical version browser displays version derivation graphs. Simultaneously a hierarchy browser displays the hierarchical decomposition.](image)

The Nelsis CAD Framework Design Management Services perform tasks such as hierarchy and version management [Wolf88a], design flow management [Bingle92a], [Bosch93a], [Wolf94a], and access control [Hoeven94a]. The Nelsis CAD Framework User Services provide a graphical interface to the engineer. These services allow
Design data and design flow management

Design data and design flow management

engineers to interact with the framework, for instance to invoke tools or to browse information stored by the framework. The services include a hierarchy browser, a version browser, (Figure 1) a design flow browser (Figure 9), and more.

2. META DATA VERSUS RAW DESIGN DATA

The central theme in our approach is that the framework maintains information about the design descriptions and the design activities, so called meta data, rather than operating at the level of the detailed design descriptions [Leuken85a]. This involves primarily information on the presence of cells, or design objects, their relationships, their version history, and the operations that have been performed on them.

A design object is the smallest object management by the framework. It contains one or more design descriptions: The design objects may, for instance, contain detailed mask descriptions, schematics, or behavioral descriptions.

The framework does not make any assumptions about the actual detailed design descriptions contained in the design objects (e.g. their granularity, or the data formats). This raw design data is operated upon by the design tools. This approach is in line with work presented by [Katz86a] and [Batory85a].

The meta data is collected by the framework while the tools communicate with it to obtain access to the actual design descriptions. A well structured meta data pool can be used to enhance design system functionality in various respects:

- The framework itself exploits the meta data to perform various services, such as versioning, concurrency / access control, or design management. To provide these services the meta data is used as the scratch-pad of the framework itself.

- Having the meta data at our disposal, smarter tools may be constructed. An example is a hierarchical verification program that exploits verification statuses from the meta data to selectively process only the design objects in the design hierarchy that have been changed since the last verification run.

- Special framework tools may be constructed which permit the engineer to interact with the framework and its meta data in order to keep track of the design process. The potential power of our approach is illustrated by the framework browsers which make the high level information about the structure and status of the design available to the engineer.

The distinction we make between meta data and raw design data helps us to distinguish between (global) system aspects and (local) tool aspects of the engineering environment, for data as well as functions. The global system aspects are implemented as part of the framework, to be offered as system-wide facilities for data organization and design process control. Thus, the framework is based on invariants that can be recognized in the engineering environment, rather than the features of a particular tool set or design representation. This is an important contribution to the openness of the environment. The distinction between meta data and raw design data is also key in achieving run-time efficiency.
3. META DATA ORGANIZATION

One of the major organizing principles in the Nelsis framework is the notion of project. A project provides a local context for the design activities of one or more engineers. It contains a collection of design descriptions that can be operated upon from within the project context. References across project boundaries are handled via an import-export mechanism, permitting projects to be used as library projects.

The meta data of a particular project describes which design descriptions are present in the project, how they are related and which operations have been performed on them. To describe which meta data has to be maintained by the framework and what its structure and properties are we have applied data modeling techniques. In contrast to work presented by [Katz86a] and [Batory85a] we have formalized the semantics of the meta data with well-defined abstraction primitives to yield an accurate definition of the information that is to be maintained. We selected the OTO-D data model [Bekke92a] to pursue our modeling activities. OTO-D stands for Object Type Oriented Data model. It is a powerful semantic data model, in which object types and their relationships can be defined, including various integrity constraints. The result of such a modeling activity is a data schema, which describes the (structural) organization of the application environment.

We have applied the OTO-D modeling techniques to formally describe such aspects as design object, hierarchy, equivalence, versioning, tool transactions, etc. The resulting data schema is depicted in Figure 2. We briefly explain the data schema; for more information on OTO-D and the data schema we refer to [Wolf88a].

With the OTO-D data model an object type is defined in terms of its attributes. An attribute relationship is graphically represented by a line that goes from the bottom of the composed type to the top of the attribute type, which may be either a base type or, again, a composed object type. Object types are represented as boxes, lines connecting corners of boxes indicate specialization and lines connecting boxes from the middle indicate attribute relationships.

The central object type in the data schema is Design Object. To a design object corresponds a design description (e.g. a schematic or a netlist) that may be accessed by the tools. A design object belongs to a module as one of its versions. Hence, DesignObject has the attributes Module, Vnumber, Vstatus, etc, to administer this module together with version-specific information such as the version number and version status. The transactions that tools perform on design objects are administered in the meta data using the object type Transaction. Design objects may be related by either hierarchical relationships or equivalence relationships. The object types ImportedDO and Export model the library mechanism, where design objects from one project can be imported (by reference) into another project.
4. DESIGN FLOW MANAGEMENT

4.1 Requirements

We provide design flow management with a maximum of functionality and a minimum impact on existing tools (executable programs). We describe the concepts of flow management in relation to a real-life framework architecture, in relation to tool interfaces and in relation to kernel framework services.

Powerful flow management in a CAD framework includes:

- Defining concepts for flow description (flow statics).
- Defining concepts for flow execution (flow dynamics).
- Proper implementation in the software infrastructure that the framework provides, such that all potential functionality of the concepts becomes available.

For incorporating flow management into a CAD system we have the following requirements:

- Incorporating flow management should not result in restrictions on the tools that can be used in the CAD system; at least the current tool set, and other similar or well known tools, should be able to function in the CAD system (including interactive tools).
• Tools should not have to perform actions specific for flow management, e.g. call special flow management functions; tool sources should not have to be modified.

• It must be possible to start tools outside the flow management user interface, e.g. from a UNIX™ command shell. In this case flow management must still be active.

4.2 Terminology and concepts

In today's engineering environments engineers use design tools to create, modify and verify their designs. During this work the primary focus of an engineer is to successfully perform design tasks. Design flow management is concerned with the execution of tools in the correct order, according to the dependencies configured in a flow description.

In our concept for design flow description we treat tools as "functional units", which can be connected by "channels" to describe data transfer. By describing the communication between tools in terms of abstract "datatypes", file details are hidden from the user. So the user may think in terms of "schematic data", "test data", etc, while data of these datatypes is actually made up of several related files. Functional units can input and output data via "ports". A port is either an input port or an output port and corresponds to a specific datatype. Channels connect ports of the same datatype to let data flow from one functional unit to another. Both branches and merges of channels are supported.

Input ports can be of type "required" (data has to be present in order to run the functional unit), "optional" (data may be used by the functional unit) or "traversal" (used to traverse a hierarchical design). Output ports can be of type "extension" (the produced data is stored in an existing design object) or "modification" (a new design object is created for storing the produced data). Distinction between extension and modification is necessary to offer a flexible concept for logical organization of the design data to the tool integrators.

An "activity" is a functional unit which can run only when data is present for all its input ports of type 'required' and which produces data for all its output ports, every time it executes successfully. Multiple activities must be defined for a tool if the input / output characteristics are different for these activities. Generally, each activity will correspond to one of the design functions of a tool. A "flow graph" is either an activity, or consists of several other flow graphs. This enables hierarchical specification of flow graphs and provides the expressive power needed to model design tasks. Examples of flow graphs

![Flow graph diagram](image-url)
are shown in Figure 3 and 4. Figure 3 shows the basic "and" "or" operation. Figure 4 shows that activity "act1" produces data of two datatypes: one is used as input by "act2" and "act3", the other is used as input by "act3" only. The two input ports of "act3" indicate that both datatypes are needed as input. "act4" needs input data produced by "act2" or "act3".

![Figure 4. Flow graph with activities (rectangles), ports (diamonds) and channels (arrows)](image)

4.3 The information model

The information model shown in Figure 5 is the model that is currently being used for design flow management. It is an OTO-D data schema and visualizes an object type hierarchy.

![Figure 5. The information model used by the framework](image)

Thus Activity is a (specialization of) FlowGraph, and Activity has a Tool as attribute, indicating the tool it is an activity of. The information model consists of parts for meta design data, flow configuration data and run-time data.

- **Meta design data** contains information about the design, such as versioning information, hierarchical relationships between different parts of the design and equivalence relationships. A detailed description of an information model for meta design data is given in [Wolf88a].
• Flow configuration data specifies the available tools, their activities and the relations between activities. The flow configuration is typically brought in by a design methodology manager before the actual design work is started. The top level flow graph is called a "flowmap".

• Run-time data is generated during the design process to correctly administer the state of design. It can be exploited to:
  • enforce the consistency of design data,
  • inform the user about the state of the design, and
  • decide what tools can or should be run (tool scheduling).

The meta design data and the run-time information are updated by the framework during each tool run according to the operations performed on the data by the tool. The flow configuration information changes only when tools or flow graphs are added, removed or modified.

5. THE CAD SYSTEM ARCHITECTURE

The CAD system architecture presented in Figure 6 has been widely accepted by standardization bodies and the industry [Liebis92a] and [CFI92a]. Therefore we believe that the approach described is applicable to other CAD systems. Figure 6 shows that tools, which may be encapsulated tools, tightly integrated tools or framework tools, are integrated on top of a CAD framework. The framework provides various services, such as data management and flow management. The framework services consult and maintain the "domain neutral data" (meta data / framework data). Design tools obtain access to the "domain specific data" (design data / tool data) via the framework.

![Figure 6. CAD system architecture](image)

Figure 6 shows the architecture of the Nelsis CAD system. As can be seen this architecture conforms to the general CAD system architecture presented above.

Additional details convey the internal structure in the Nelsis CAD system. Flow management is a service in the framework kernel, and not part of a desktop or other framework tool. This ensures that tools cannot simply by-pass flow management. Domain neutral data is stored in a special purpose lightweight DBMS that can be configured with an OTO-D data schema, such as the information model in Figure 5. The DBMS provides a SQL-like query interface for convenient data retrieval and manipulation. The domain specific data is currently stored in the UNIX™ file system.
Design data and design flow management

thereby providing efficient and compatible storage for many of the file based design tools. Domain specific data is organized as design objects. A design object is a version of (a part of) the design and contains data of a certain view type. E.g., a design object may contain schematic data of a flip-flop. Per design object data is organized as one or more data "streams" that may, for example, correspond to design files.

The framework services have the following functionality:

- **Flow Management:**
  - *Activity Engine:* Checks the data access operations of a tool against the tool’s configured activities. Identifies the activity the tool is performing.
  - *Flow Engine:*
    1. Determines the internal state of design objects (availability / validity of data in design objects).
    2. Checks whether activities are allowed concerning: (1) input data and (2) the flow configuration. That is, it determines whether the producer of the input data is a valid producer for the activity being performed.
  - *Transaction Manager:* Administers activity runs and transactions describing the use of design objects in an activity run.

- **Data Management:**
  - *Lock Manager:* Locks and unlocks design objects for coordinated multi-user access. It prohibits conflicting operations, such as multiple concurrent write operations, on the level of design objects.
  - Other: E.g. configuration management, hierarchical consistency management and version management.
Tools interact with the framework via the Data Management Interface (DMI, a procedural interface) to obtain access to design data. Tightly integrated tools perform calls to the DMI functions in the source code; for encapsulated tools a 'wrapper' program performs the necessary DMI calls. By offering a proper set of "anchor points", the standard DMI greatly facilitates software exchangeability and permits framework and tools to evolve separately to a large extent. The DMI does not incorporate any specifics for purposes of flow management. Being a pure and simple interface geared towards data access only, the framework services are hidden from tools. The calling pattern in Figure 8 illustrates how the DMI functions cooperate.

```
DM_PROJECT projectkey;
DM_DESIGNOBJECT desobjkey;
DM_STREAM streamkey;

dmInit (toolname);
projectkey := dmOpenProject (projId, mode);
desobjkey := dmCheckOut (projectkey, desobjld, mode);
streamkey := dmOpenStream (desobjkey, streamId, mode);
dmPutDesignData (streamkey, format, arguments);
dmGetDesignData (streamkey, format, arguments);
dmCloseStream (streamkey, mode);
dmCheckIn (desobjkey, mode);
dmCloseProject (projectkey, mode);
dmQuit ();
```

**Figure 8. DMI calling pattern**

6. RESULTS

The design flow management of the Nelsis CAD framework has been completely transparent to the tools since design flow management has been implemented "underneath" the standardized DMI. The DMI has currently been stable for over 8 years. During this time the framework implementation evolved from a simple single-user, single-host data management system, that was little more than an abstract file system, to the powerful multi-user fully distributed framework that it is today, relieving engineers from many burdens and directly helping them to concentrate on their main task: design. This clearly proves the DMI paradigm.

Since the framework keeps track of the data accesses performed by tools, activities can be recognized while the tools are running. Thus the modified internal state of design objects can be displayed before tools finish execution. This is of special interest for long running interactive tools such as, editors or interactive simulation environments, with which engineers may perform multiple design actions during a single tool run.

Based on the implemented flow management components we have implemented a graphical flow browser that visualizes the state of the design. The flow browser obtains the state of design objects from the flow engine. It displays the existence of data on input and output ports and highlights the flow graph instances, using colors or line styles, to reflect the flow status. Using the mouse, engineers can start activities and zoom into hierarchical flow graphs. The flow browser can also start activities automatically and give advice on tool scheduling based on the flow configuration data. The flow browser is
part of the Design System User Interface [Leuken95a], the central graphical user interface of the Nelsis CAD Framework, which integrates information retrieval, design object selection and tool activation in a coherent user interface. Figure 9 shows a window dump of the flow browser displaying status information on version number 3 of a design called 'rand_cnt'.

Figure 9. Flow browser window

Thick lines and filled ports indicate that data is present, a dotted box indicates that a flow graph is not executable, a dashed box indicates that a flow graph is executable and a solid box indicates that a flow graph has executed. Figure 9 shows that the box simulate is a compound. In simulate the program 'sIs' has been executed, and the program 'simeye' can be executed. In the flowgraph root, the program 'match' cannot be executed because not all input data is available.

7. CONCLUSIONS AND FUTURE WORK

The Nelsis CAD Framework has been implemented in the C-language on the UNIX™ operating system. The graphical framework tools, such as the flow browser in the DSUI, make use of OSF/Motif™ and the X Window System™.

Evaluations of the implementation have shown that the ideas do indeed provide a fully functional design system that guides engineers through the different design steps, shows them the state of the different parts of their design and enables convenient automatic and
manual activation of design tools.

The user community of the Nelsis CAD Framework includes IMEC (Belgium), University of Southern California (USA), CRS4 (Italy), GMD (Germany), Philips (NL), and others. The application areas are among other: automated software testing, semiconductor design (synthesis), climate studies and image processing. In general the Nelsis CAD Framework is used for system design and system simulation environments. Whereas these users have reported positively about the effectiveness of the technology provided, they have also expressed the need for the 'next level of automation' to support coordination and communication among engineers in the larger engineering projects. Research and development activities in this area have started.

References


The Concept of a Design and Planning Platform

Atsumi Ryo
Yachiyodai Minami 2-6-7, Yachiyo Shi, Chiba, 276 Japan
Telephone +81-474-82-1729, Japan
Facsimile +81-474-82-1729, Japan (only after oral call)

Abstract
Various models have been proposed for design and planning activities. However, it is difficult to grasp their individual total images because their positions and meanings in actual design or planning activities are not necessarily made clear. The author has carried out studies to clarify the individual positions of these design and planning activity models and to understand their structures and behaviors.

Furthermore, the author, based on analyses, proposes to systematize various design and planning activity models into a tool called a design and planning platform. This tool will be included into a social human activity model proposed by the author based on the concept called the natural division of labor. This concept insists that the distribution of authority and responsibility should be the principle.

Keywords
Design, planning, deployment, evaluation, internal structure, business process, human protocol

1 Introduction

Recently, such concepts as intelligent activities, distributed and autonomous systems, and organizational intelligence\(^\dagger\) have attracted the attention of the world. This indicates that the current trend is to focus on both human activities and man-machine systems which includes social human activities. And an integrated understanding of these two is being requested. Integration represented by the OSI reference model for communication consisting of 7 layers (ISO 1984) is being actually developed. However, such frameworks are mostly confined to those for machines, and a framework for human activities and man-machine systems, comparable with those for machines, has not appeared yet.

\(^\dagger\) Organizational intelligence is the title of a session at The International Conference on Economics/Management and Information Technology (CEMIT) '92, and is also the name of a new paradigm being proposed by Dr. T. Matsuda, former presidents of the Tokyo Institute of Technology and the Sanno Institute (and also former V.P. of IFORS), becoming the object of attention in recent years (Umibe, 1991)
The author considers that this trend mentioned above can be summarized as an organism-oriented trend which has spread out into numerous branches. These branches can be classified into four categories: biological models, pseudo-biological models, pseudo-human activity models, and human activity models, and the approach by a human activity model is most promising at present. Among the human activities which can be classified into philosophical, social, and biological aspects, the understanding of social human activities is most expected, because social human activities are exactly those human activities and man-machine systems mentioned above.

The author has already proposed a framework for social human activities named a social human activity model obtained by the study of actual social human activities (Atsumi, 1993a). This social human activity model is expected to be able to integrate the domains and hierarchies of various social human activities by providing interfaces for them, because it is based on social human activities.

However, the proposed model up till now has left design and planning activities unsettled. Although various models have been proposed for design and planning activities, it is still difficult to grasp their individual total images because their positions and meanings in actual design and planning activities have not necessarily been made clear. Consequently, some models will be analyzed for their integrated understanding and the feasibility of actual applications through making clear their positions, structures, and behaviors.

2 Social Human Activity Model

The author has already proposed the concept called the natural division of labor which insists that the distribution of authority and responsibility should be the principle and centralization should be limited only when it is proven to be effective (Atsumi, 1993a). Here, the activity unit of social human activities is each modular unit of the division of labor, i.e. the unit exchanging orders, which is called an order cell, and the external function of which is defined as exchanging orders with each other, as shown in Figure 1. The internal function is defined as having a set of missions which are unfinished orders and a set of all the functions necessary to carry them out, and has the form of processing orders. And, it asserts that social human activities can be represented by a hierarchy of order cells, which is an expression of a social human activity model in a broad
The concept of a design and planning platform

sense and named an order model. The order model employs orders as unique objects to be processed and consist of units of a single unique structure for processing such unique objects. It is applicable to all the necessary functions extending from management to services for all layers from a public or private organization to an equipment control device.

Furthermore, the author has already proposed an operational internal structure (Figure 2) of an order cell in which the human protocol based on the principles of social human activities presenting a framework is located on one end, and theoretical models based on the principles of science giving logical meaning on the opposite end, and various applications are assumed to exist between them. This structure is named an order cell model, which expresses a social human activity model in a narrow sense.

Various concepts have currently been proposed for computer integrated manufacturing systems, such as the Purdue Enterprise Reference Architecture (Williams, 1994), the Integrated Enterprise Modelling (Mertins, 1991), and the CIMOSA Model (AMICE, 1993). However, they are all top-down models and instruct very few about the internal structure of individual business processes. This is very disappointing to a practitioner. A simple and unique internal structure applicable to all business processes including design and planning activities is indispensable for making such enterprise models easy to apply. Also, the so-called work flow system, concurrent engineering, or continuous acquisition and life cycle support (CALS) is not an exception for the lack of an internal structure of individual business processes. The author insists to focus on the internal structure of business processes.

3 Human Protocol

The human protocol (Atsumi, 1992 & 1993a) has a multilayer structure following the 7 layer OSI reference model for communication (ISO 1984). The real world for the multilayer structure of the human protocol is considered to be that of office work by humans, by which the author has proposed such layers of process categories consisting of operation, work, procedure, strategy, and goal. The basic principle throughout the above categories is explosion, and orders are exchanged between the above layers, and processed by them. The procedure for constructing individual layers, based on the principle of focusing on human experience, is to select the representing real world, to establish its corresponding model, and to construct the operating system.

The real world for the first layer is operations on the desk, and the model is office automation (OA). The user interface is an operational user interface (OUI), represented by the well known graphical user interface (GUI), and the operating systems are various existing window systems.

For the second layer, the real world is office work by handwriting, and the model is business work. The user interface is a clerical user interface (CUI), and the operating system is a worksheet program (WSP) (Atsumi, 1991a & b, 1993b).

For the third layer, the real world is production control, and the model is order processing control. The user interface is a planning user interface (PUI), and the operating system is order processing control (ODPC) (Atsumi, 1993c).

For the fourth and fifth layers, the author focuses on design activities as the real world. Design activities, which can be considered as a kind of planning activities, are a major portion of intelligent activities including planning activities performed in public and private organizations, and have been processing a large mass of information in routine work. By due to this reason, various processing methods have been developed and practiced in design activities.
Quality function deployment and general design theory

Quality function deployment by S. Mizuno and Y. Akao (Mizuno, 1978, Akao, 1990) is discussed in order to prepare a tool for analyzing various design and planning activity models. Quality function deployment has its origin in the quality control activities of Japan. Those people who were not satisfied in merely achieving given quality standards, traced back to the origin of the quality standards and went into the domain of product design, and furthermore into the domain of product planning. Their activities have developed quality function deployment into an integrated design method which extends from quality planning and design to work standards in the shop.

Quality function deployment is constituted by:
- quality deployment in a narrow sense,
- technology deployment,
- cost deployment,
- reliability deployment,
- and quality function deployment in a narrow sense.

Quality deployment in a narrow sense, which is the most essential among these five constituents, consists of:
- demanded quality deployment and quality planning,
- quality characteristic deployment and quality design,
- parts deployment,
- and process deployment.

Quality deployment in a narrow sense, which converts the degrees of importance of the consumers' demands to those of the component parts using matrix, has been systematically established and has the possibility to be the nucleus of various design and planning methods. Accordingly, the author has tried to extend the concept of quality deployment.

T. Ohfuji\(^2\) has an idea to generalize quality deployment by deleting the word "quality" from "demanded quality" and changing it to "demands." According to this idea, the word "quality" will be completely deleted from the description of quality deployment.

Also, constraint deployment, which consists only of technology deployment, cost deployment, and reliability deployment up till now will be extended to include all the constraints of the product aspect, organization aspect, and social aspect.

Furthermore, the general design theory proposed by H. Yoshikawa (Yoshikawa, 1979) models design activities as mapping from a function space to an attribute space. In his general design theory, an entity is defined as a matter which exists, existed, and will exist, and an attribute is defined as various physical and scientific characteristics, and behavior is defined as the change in an attribute with time. Function is defined as those characteristics abstracted by applying a certain rule to an attribute and behavior. Such definitions will be extended so that design and planning activities are defined as mapping from an original set to an object set. They include those activities generating all or a part of an original set, object set, or mapping function for applying the design activity model of the general design theory to design and planning activities.

By such an extended general design theory, extended quality deployment can be defined as a downward design and planning activity model, which generates an original set: (demand deployment), determines the type of mapping function: (characteristic deployment), establishes a mapping function: (characteristics), maps an original set: (demand deployment) to an object set: (parts deployment) generated beforehand. Anything can be

\(^2\) Tadashi Ohfuji: Assistant Professor, Faculty of Engineering, Tamagawa University
included into demands unlimitedly. Concepts, processes, and actions can be included into parts since parts are extended to include documents/information, work/services in addition to the so-called mechanical or physical type of parts. Furthermore, a mapping function can be regarded as a scenario.

5 Integrated Model

The following methods, which can integrate a series of design or planning activities, will be analyzed.

5.1 Soft Systems Methodology

P. Checkland has proposed Soft Systems Methodology (SSM) as a tool for the perpetual improvement of a human activity system. He has defined SSM as “a set of activities linked together in a logical structure to constitute a purposeful whole.” (Rosenhead, 1989).

Soft Systems Methodology can be considered to be a method which

- hierarchically generates root definitions: (demands and demanders),
- builds a conceptual model: (a combination of demand deployment, characteristic deployment, constraint deployment, and parts deployment),
- analyzes by comparing the model with reality, and defines changes.

The value of this method lies in the fact that it hierarchically designs or plans downwards or upwards, and finds out a culturally feasible solution by laying emphasis on the persons concerned, their power, norms of behavior, and the basic cultural values.

5.2 Strategic Choice Approach

J. Friend has proposed a procedure to deal with uncertainties, insisting that the routing of any non-routine decision process is governed by the perceptions of three types of uncertainties; they are uncertainty pertaining to the working environment, uncertainty pertaining to guiding values, and uncertainty pertaining to the related decision fields (Rosenhead, 1989).

The strategic choice approach can be considered to be a method which

- deploys the decision areas: (characteristic deployment),
- extracts available options: (might-be characteristic values),
- identifies feasible decision schemes: (combinations of parts deployment and characteristic values, that is, scenarios),
- compares feasible decision schemes: (determination of the degrees of importance) for comparison areas: (demand deployment),

and then, using the results derived above,

- identifies and compares uncertainty areas for comparison areas: (demand deployment and planning),
- decides present actions and explorations promising an advantageous future decision space: (scenario regarding future decision space, present actions, and present explorations as characteristics).

This approach intends to control uncertainty through the process of two stages consisting of the first stage clarifying the structure of the problem upwards, and the second stage making decisions downwards.

5.3 Robustness Analysis

J. Rosenhead has proposed to cope with uncertainty by using robustness, regarding planning under certainty as sequential decisions, that is, planning with uncertainty consisting of present decisions and the remaining future (Rosenhead, 1989).
The robustness of any initial decision is defined to be the number of acceptable options of the planning horizon with which it is compatible, expressed as a ratio of the total number of acceptable options at the planning horizon.

Robustness analysis can be considered to be a method which
- generates a tree of decisions: (candidate characteristic values) of multi-stage: (characteristics) for the target situation: (parts deployment),
- values: (planning) options: (parts compositions) for individual targets: (demand deployment),

and then, by using the results derived above,
- makes the decision: (characteristic values) of the present state: (characteristics) expected to be more flexible and desirable selections in the future: (demands).

This analysis is a method similar to the one described in paragraph 5.2 using a two stage process consisting of the first stage clarifying the structure of a problem upwards, and the second stage making decisions downwards, but different from the one described in paragraph 5.2 in that it simplifies evaluation by using the number of options.

5.4 Metagame Analysis
N. Howard has proposed to apply metagame analysis which takes strategy and human relations into consideration to a situation in which individual actors choose their options in order to win against others (Rosenhead, 1989).

This metagame analysis can be considered to be a method which
- clarifies the situation by preparing a list of actors: (parts deployment) and their options: (parts characteristic values),
- finds feasible scenarios: (characteristic values) by combining the options,
- draws a strategic map, a diagram of transitions among scenarios: (stability analysis of characteristic values) by finding out changes of scenarios preferred by actors and countermeasures against such
- changes through the preferences of scenarios of individual actors,

and then, using the strategic map,
- finds the conditions: (characteristic values) to realize the scenario: (demands) preferred by a certain actor by introducing preference change, irrationality, deceit, or rational argument in the common interest: (characteristics).

This analysis intends to process the situation of decision making by multiple actors, considering it as a state transition, by using a two stage process consisting of the first stage clarifying the structure of the problem upwards, and the second stage making decisions downwards.

5.5 Hypergame Analysis
P. Bennet, et al., have proposed to apply hypergame analysis to the analysis of the world of decision making by multiple players individually, which presumes that the players concerned holding different views of the situations make decisions affecting the well-being of others for the purpose of advancing their own aims and interests under a pluralist view of the world (Rosenhead, 1989).

The above presumptions lead to a basic hypergame model, in which there are separate games for individual players who specify the strategies, preferences, and presumed results of all players concerned. Furthermore, the model is expanded to include radical differences in the perceptions among players. These differences consist of the definitions of the games in quite different terms, disagreement of the extent of the relevant parties, and disagreement about the authorship of actions.
This hypergame analysis can be considered to be a method which

- identifies players: (parts deployment), finds out options: (parts characteristic values),
- selects scenarios: (characteristic values) through the elimination of infeasible solutions, stability analysis using the order of preference, and analysis of the implementation sequence,

and then, using the results obtained above,

- structures the hyper game models,
- analyzes to obtain the preferred result: (demands),
- and new ideas: (characteristics) are introduced if a preferred result is not obtained.

This analysis intends to apply the hypergame analysis having separate games for individual players to the world of decision making by individual multiple players by use of a two stage process clarifying the structure of the problem upwards, and the second stage making decisions downwards.

The description for analyzing individual players by using a view of the world: (demand deployment), actions: (parts deployment), options: (parts characteristic values), scenarios: (characteristic values), and decision areas: (characteristic deployment) according to the extended quality deployment will explicitly express the world of pluralism to make the analysis more easy.

5.6 Taguchi Method

G. Taguchi has proposed quality engineering (Taguchi, 1986 & 1988), which determines the parameters for design, tolerances based on the concepts to maximize the signal-to-noise ratio (S/N ratio), and to minimize loss and to balance quality and cost.

The S/N ratio is defined as the ratio of the mean response value by a controllable factor against the variance of responses due to uncontrollable factors. Quality is defined as the mean actual economical loss (the loss function), regarding quality as the loss to customers.

The Taguchi method can be considered to be a method which

- determines the level of factors: (characteristic values) based on the S/N ratio,
- and evaluates the adequacy of the selection of factors: (characteristic deployment) by the amount of interactions among the factors in parameter design,
- determines the parameters for design: (characteristic values) according to the principle of minimizing overall loss: (characteristics), and the tolerance: (characteristic values) by balancing quality and cost: (characteristics) in tolerance design.

This method provides an effective tool for processing the successive characteristic deployment, and is noticed in that it evaluates the adequacy of the characteristic deployment itself.

5.7 Metamodel Mechanism

Kiriyama, et al., have proposed to integrate various design object models for different physical aspects. This is based on the knowledge of the relations among the concepts which are used in the theories expressing physical aspects, which is named a metamodel mechanism (Kiriyama, 1991).

The metamodel mechanism can be considered to be a method which automatically transforms

- a primary model: (demand deployment) into a central model: (characteristic deployment and characteristic values),
- a central model into various aspect models: (parts deployment considering their individual aspects and their details as component parts).
<table>
<thead>
<tr>
<th>Extended quality Design Deployment Method</th>
<th>Demand deployment</th>
<th>Planning</th>
<th>Characteristic deployment</th>
<th>Design/Characteristic values</th>
<th>Parts deployment</th>
<th>Parts design/Part characteristic values</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soft Systems Methodology</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Strategic choice approach</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robustness analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metagame analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hypergame analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modified hypergame analysis</td>
<td>A view of the world</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Taguchi method</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metamodel mechanism</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3 Comparison of Integrated Models for Design and Planning.

Remark: ①, ②, ③ ... indicates the operation sequence.
This method can be generalized to apply it to a world in which logical mapping is possible, and should be applied as much as possible.

5.8 Comparison of Methods
A comparison of the individual methods with their respective processes for extended quality deployment is shown in Figure 3.

Figure 3 shows the following.
- The individual methods should not be discussed for the purpose of evaluating which are better, but should be selected for applying them according to the issue, object, or aspect, since they have their unique respective focuses.
- Extended quality deployment is effective for analyzing design and planning models.
- Fixed items shown with "...." in Figure 3 suggest the possibility of standardizing these items.

6 Design and Planning Platform

The fourth layer, "strategy", and the fifth layer, "goal", described in Section 3 will be studied based on the discussions in Section 4, and 5. To separate strategy and goal is irrational, as it can be seen from the case of a systematic diagram (Mizuno, 1988), which regards the means or procedures of the preceding step as goals or objectives. Accordingly, the fifth layer "goal" will be eliminated, considering input as the goal, and output as the strategy. The world of design and planning activities is selected as the real world for the fourth layer.

The basic processes of the real world are to generate all or a part of an original set, object set, or mapping function, and to map the original set to the object set according to the extended general design theory. These processes can be considered to be one or two dimensional deployment and its evaluation, as illustrated in Figure 3. They are to extract elements, to find interrelations among them, and to express the intention. Consequently, the model is a deployment and evaluation process (DEP).

As for the operating system, it seems impossible to adopt a unique method for the operating system, since actual design and planning activities consist of various combinations of various deployment and evaluation unit processes having interfaces to others, as shown in Figure 3. In the evaluation of methods, it is not sufficient that the method is equipped with many and powerful functions. It should rather be evaluated from whether it is easy and simple to use or not and whether it is applied widely or not and also whether the effectivity of the method itself can be increased by the advancement of the users' skill

<table>
<thead>
<tr>
<th>Layer</th>
<th>Process</th>
<th>Real world</th>
<th>Model</th>
<th>User interface</th>
<th>Operating system</th>
</tr>
</thead>
<tbody>
<tr>
<td>4th</td>
<td>Strategy</td>
<td>Design and planning</td>
<td>Deployment and evaluation process</td>
<td>Strategic user interface</td>
<td>Design and planning platform</td>
</tr>
<tr>
<td>3rd</td>
<td>Procedure</td>
<td>Production control</td>
<td>Order processing control</td>
<td>Planning user interface</td>
<td>Order processing control</td>
</tr>
<tr>
<td>2nd</td>
<td>Work</td>
<td>Office work by hand writing</td>
<td>Business work</td>
<td>Clerical user interface</td>
<td>Worksheet program</td>
</tr>
<tr>
<td>1st</td>
<td>Operation</td>
<td>Operation on desk</td>
<td>Office automation</td>
<td>Operational user interface</td>
<td>Various window systems</td>
</tr>
</tbody>
</table>

Figure 4 Structure of a Human Protocol.
through actual use or not. So, a method which appears rather primitive might be more useful. Furthermore, a diversity of methods should be provided to cope with complicated reality. Consequently, for an operating system, the author proposes a design and planning platform (DPP), which presents available methods, applicable fields, application limits, features, texts, manuals etc., for the individual stages of design and planning activities. A computer environment including tools and standardized item lists etc. necessary to apply the presented tools has also been provided.

Furthermore, design and planning activities have been brought under the support of the third layer represented by material allocation and sequence control, the second layer represented by document control, and the first layer represented by office automation through the incorporation of these activities into the fourth layer of human protocol. In other words, any consideration has become unnecessary for the third and less layers by their incorporation, whereas it should be noticed that the lack of consideration for the infrastructure mentioned above is the common deficiency throughout the analyzed methods. Namely, the desired result cannot be obtained through one trial, but the result will be gradually refined to the desired one through repeated trials in the practical use of these methods. Also, changes in design or planning are inevitable in all aspects of any applications to practical design or planning activities. Furthermore, the results of any design or planning activities become proofs for such matters as product liability, records for maintenance, and accumulation of experiences for reuse. Tools not providing such an infrastructure will result as a burden on their users.

In addition, there exist problem analyzing models used for making a structure, one dimensional deployment models used for expressing relations by distance, and two dimensional deployment models used for expressing relations by a matrix applicable in the individual stages of design and planning activities. And these models should be incorporated into the design and planning platform.

A problem analyzing model connects the nodes of ideas or thoughts with directed links. Cognitive mapping (Tolman, 1948) and relations diagram (Mizuno, 1988), both of which link the elements with logical relations, and an affinity diagram (Kawakita, 1970, Mizuno, 1988), which links the elements with mental relations, belong to this category.

A one dimensional deployment model expresses problems or circumstances with multiple items and distances among them. A systematic diagram (Mizuno, 1988) and process decision program chart (Mizuno, 1988), both of which locate items according to a family type logical distance, and an articulation aid system (Hori, 1994) and common quality demands (Nakajo, 1994), both of which locate items according to the mental distance, belong to this category.

A two dimensional deployment model generates a relation matrix between a pair of the results of one dimensional deployment. A matrix diagram (Mizuno, 1988), which is the most simple method, matrix data-analysis (Mizuno, 1988), which uses principal component analysis, and near decomposition of quality table (Shindo, 1993), which decompose the matrix into partial matrixes by the quantification method of type 3'', belong to this category.

Above discussion gives a structure shown in Figure 4 to the human protocol.

7 Conclusion

The author has carried out studies to establish the fourth and fifth layers of the human protocol by clarifying design and planning activities and structuring them. These represent the internal structure of one of the social human activity models, i.e. the order cell model.

For this purpose, the author has extended the general design theory and quality function deployment, has analyzed various design and planning methods using extensions of these
methods, and has obtained the deployment and evaluation processes as a unified principle for these methods.

Consequently, the author has proposed to eliminate the fifth layer of the human protocol, and to make the real world, the model, and the operating system of the fourth layer, strategy, as design and planning, deployment and evaluation process, and design and planning platform, respectively.

There exist several discussions on design, planning and production control that they are different tasks of the same level. However, the author's interest only lies in their structures, and insists that design and planning have the same internal structure but production control has a different internal structure. Here, it should be noticed that the structure of production control is indispensable for design and production activities as their substructure, and planning required for production control can be executed on a design and planning platform.

The issues for further studies are to select methods and to prepare menus, tools, and standardized item lists for the individual stages of these design and planning activities. Furthermore, the suggestion of the possibility of standardizing within the individual phases of the extended quality deployment stated at the last of Section 5 should noticed. It implies the existence of internal structures in the individual phases. These structures should be studied before the practical application of the design and planning platform.

Also, most concepts currently proposed for computer integrated manufacturing etc. are attempting to directly describe complicated actual business processes or activities, and are being confronted with difficulties. The concepts proposed in this paper suggest that these difficulties are caused by neglecting the study of the structure of business processes. Namely, the author considers that the basic practice of business processes and their principles should be studied just like the Ohm's law in electricity. These principles should be systematized to establish a new technique, which can be called business process engineering.

8 Acknowledgments

This research was performed as a domestic feasibility study of a joint international research program on intelligent manufacturing systems (IMS). The author wishes to thank Mr. Fujio Umibe, a consultant, for reviewing the original manuscript and suggesting revisions in its English.

9 References

and J. Goossenaerts), IFIP Transaction B-14, Elsevier Science B.V. (North Holland).

(under examination).

08/084,066 (allowed issuance).


Kiriyama, T., Tomiyama, T., Yoshikawa, H. (1991), "The Use of Qualitative Physics for

Mertins, K., Siissenguth, W., Jochem, R., (1991), "Integration Information Modelling,
Proceeding of 4th IFIP Conference on Computer Application in Production and
Engineering (CAPE '91).


Press.

Quality Requirements: A Case Study on the Products Used by University Students (in
Japanese)," Journal of the Japanese Society for Quality Control, Vol. 24, No.1, pp.101-
109.


Shindo, H. (1993), "Generalization of Quality Table Concept as a Method of System
Description and its Structuralization Based on Near Decomposability (in Japanese),"


208

Umibe, F., (1991), "Some Comments and Proposals for Better Application of Theories and
Tools to Actual Practise", IEEE Transactions on Engineering Management, Vol. 38,
No. 4, pp.359-364.

Vol 24/2-3.


10 Biography

Mr. Atsumi Ryo, born in 1939, master of mechanical engineering, has been interested in
developing his social human activity model to overcome post-structuralism. He is at present
engaged in the assessment of ISO9000 quality systems. He is an old practitioner engaged
in system engineering (automated design, production process control, profit control, promo-
tion of engineering work computerization, education and management of software forces),
establishment of a nuclear quality assurance system and a power boiler production system,
design of chemical process plants, project engineering of chemical process plants and power
boilers, and procurement of cast and forged products.
EXPRESS definition of Vectorial Tolerancing in product modelling

Kristian Martinsen, Toshio Kojima
Mechanical Engineering Laboratory
Namiki 1-2, Tsukuba-shi Ibaraki-ken, 305 JAPAN
tel: +81 298 58 7013, fax: +81 298 58 7007
Email: kojima@mel.go.jp

Abstract
Tolerance information is essential for manufacturing processes, because all processes will have a certain variation. It is beneficial to treat the tolerance data in a unified manner in product modelling systems. For this purpose, we took an approach to describe tolerance information of Vectorial Tolerancing (VT) in a formal way using EXPRESS developed by STEP. In VT, surface location and orientation is described with vectors in a Workpiece Coordinate System. VT provides a clear distinction between the size, form, location, and orientation deviations with magnitude and direction and is therefore useful for functional analysis and the control of manufacturing processes.

Keywords
Vectorial Tolerancing, EXPRESS, STEP, Product Modelling

1 INTRODUCTION

Manufacturing processes will always have a variation. The function of a product, the chosen manufacturing processes and thereby also the manufacturing costs are dependent on the tolerances. So, it is important to develop a tolerancing model which can express the functional relations as well as manufacturing considerations in a unified manner in three dimensional product modelling systems.

The objective of this paper is to define a formal description of Vectorial Tolerancing (VT). This will be a basis of the future information infrastructure system to realize a fully integrated intelligent manufacturing system. The definition is done in EXPRESS, which is developed in order to define product model data in a formal manner (ISO 10303-11, Schenck, 1994). This
makes possible a clear comparison to ISO tolerances and other tolerancing methods. The EXPRESS definition can also be used for implementation of VT in product modelling systems.

The existing ISO tolerances are to some extent ambiguous and need human interpretations as they are based on one-dimensional measuring. ISO tolerances on orientation, location and size are sums of different tolerancing features where the magnitude of each feature is unknown. ISO defines for example a size of a cylinder as a distance between two points. The two-point distance will be influenced by the local form deviation in each of the two points as well as the orientation deviation of the cylinder. A measured distance between two planes, on the other hand, will be a sum of the form, orientation, and location deviation on each surface. Tolerance ambiguity occurs because the ISO-tolerance does not define any measuring direction for the distance tolerance. Orientation tolerances are in ISO defined as a one dimensional distance between two enveloping surfaces, see Figure 1. The orientation measurement is because of this unable to give information about the direction of the orientation deviation. Corrections of orientation deviations in the manufacturing process are therefore impossible. Furthermore, because form deviations are included in the ISO orientation tolerance, there is no information about how much of the orientation deviation is caused by deviation of form.

![Figure 1. Orientation deviation, Comparison of ISO and Vectorial Tolerancing.](image)

Many new methods are proposed to implement tolerances in product modelling systems. Requicha (1983) uses tolerance zones that cover all types of deviations in form, size, orientation and location. This means that the ISO tolerances of flatness, etc. have become superfluous. Other authors (Farmer and Gladman 1986, Weill 1988) have argued that this definition takes no account of functional, manufacturing, assembly and measuring requirements. Jayaraman and Srinivasan (1989) describes a method where tolerances are given as the theoretical limit boundaries when the combined effects of all associated tolerances are taken into account. This approach assures interchangeability of parts in assembly. J.U. Turner et. al. (1990) and J.U. Turner (1993) suggested a tolerancing theory where tolerance specifications are interpreted as constraints on a normal vector space of model variations, i.e., a vectorial representation of all ISO geometrical and dimensional tolerances constraining the actual surface. Technologically and Topological related Surfaces (TTRS) was introduced by Clement (Dufosse 1993, Gaunet 1993, Clement 1993, Clement 1995 and Desrochers 1995). In TTRS the association of two surfaces or group of surfaces is described with the 6 degrees of freedom (DOF). There are therefore 6 independent tolerancing parameters. Clement defines a Minimum Geometric Datum element (MGDE) which is the minimum set of point, line or plane necessary and sufficient to
define the displacements. Both VT and TTRS are based on the six degrees of freedom, however, VT uses location and orientation vectors and TTRS using the six degrees of freedom directly. TTRS implies using a tangential surface as the Substitute surface while VT is using best fit substitute surface. Separation of form deviation is therefore not possible in the TTRS model. The TTRS model are arranging surfaces and surface groups in pairs, while VT are using Workpiece Coordinate Systems.

In the following sections, we will introduce constructs of shape data in VT one by one in EXPRESS. Only the essential of the EXPRESS code is given here, using plane and cylinder as examples. The other types of surfaces: general, prismatic, spherical and rotation symmetrical surfaces, will not be shown here. This grouping of surfaces is according to the open DOF for each type of surface and is almost identical to the TTRS groups. (Dufosse 1993, Gaunet 1993, Clement 1993, Clement 1995 and Desrochers 1995)

2 VECTORIAL GEOMETRY MODEL

Vectorial Tolerancing (VT) was introduced by Wirtz (Wirtz 1988, Wirtz et al. 1990, Wirtz 1991, Wirtz et al. 1992). A similar model is proposed as a new ISO standard; new work item to ISO TC 10, ISO TC 3, dimensioning and tolerancing. This is also described by Henzolt (1993). Martinsen (1993) showed VT can be used for all types of surfaces. VT is a mathematically unambiguous model for describing nominal vectorial geometry and tolerances. It is suited for computer representation in the link between design, manufacturing, measuring and process control. VT utilises 3D tolerances as an integrated part of the Vectorial product model with automatic generation of NC-codes and measuring programs (Wirtz, et al., 1993). VT provides a clear distinction between size, form, location and orientation for each surface. Such a distinction is important for both functional analysis a manufacturing process control (Martinsen, 1995).

2.1 Degrees of freedom

The principle of 6 degrees of freedom (DOF) is a basic concept for VT. An infinite plane can be translated in two directions and rotated in one without changing the location and orientation of the plane. This is called the open DOF for a plane. An infinite plane has 3 open DOF, and therefore 3 fixed DOF. Only the fixed DOF of a surface needs tolerances. In the case of a plane, the position of a plane is tolerated by one tolerance on location, and two on orientation. An infinite cylinder can be translated in the axis direction, and rotated around the axis without changing the location and orientation of the cylinder. Hence, a cylinder has 2 open DOF, and needs tolerances on two location components and two orientation components, perpendicular to the cylinder axis.

2.2 Vectorial surface description

A surface is described by its location, orientation, form and size. In VT, location is described with a location vector, and orientation with a unit orientation vector. A nominal surface has a nominal location vector and orientation vector which describes the nominal position of the
surface. On a real workpiece, the real location, orientation and size are defined by a geometrically ideal substitution surface with the same form as the nominal surface. The substitute surface is calculated by the Gaussian best fit algorithm from the assessed coordinate measuring points on the real surface. Figure 2 shows the VT model of a plane. Deviation vectors are the subtraction of the nominal from the real vector. The location vector fixes the translation DOF of a surface. For surface types such as torus and sphere, the location is given as the centre point of the surface. For surfaces that do not have any well-defined centre point, the point has to be chosen. For cylinders, a point on the cylinder axis is chosen, and for a plane a point in the plane. This should, however, be done with a view to the function the surface will have in the assembled product. The orientation vector is a unit vector which fixes 2 rotational DOF of a surface. The orientation vector of a cylinder is the direction of the axis and for a plane the normal vector.

Figure 2. VT model of a plane.

ENTITY location_vector;
name : STRING;
x,y,z: length_measure_with_unit;
END_ENTITY;

ENTITY orientation_vector;
name : STRING;
x,y,z: REAL;
WHERE x**2 + y**2 + z**2 = 1 ; -- Length = 1
END_ENTITY;

ENTITY length_measure_with_unit;
unit : STRING; -- e.g. mm.
value : REAL;
END_ENTITY;

In addition to location and orientation, a surface has a form and size. The form of the surface is described with an equation (or a set of parametric equations) with sizes as parameters. The sizes tells the size of the surface. It can be the radius of a cylinder or the apex angle of a cone. A torus will have two sizes, one small and one large radius. The corresponding size of a surface is by definition negative for inner surfaces and positive for outer surfaces. Other surfaces can have several sizes. The form deviations are in VT filtered out by the best fit algorithm. The form feature can then be treated separately.

The following is the EXPRESS definition of surfaces in VT. The definition of tolerancing attributes is described in section 3. Every surface is defined in a coordinate system, which is discussed in section 2.4.
Vectorial tolerancing in product modelling

ENTITY vt_face_surface ABSTRACT SUPERTYPE OF (ONEOF (general, prismatic, rotational, cylinder, plane, sphere));
  name : STRING;
  basis : coordinate_system;
END_ENTITY;

ENTITY plane SUBTYPE OF (vt_face_surface);
  nom_location_vector: location_vector;
  nom_orientation_vector: orientation_vector;
  z_location_tolerance: location_vector_component_tolerance;
  x_orientation_tolerance, y_orientation_tolerance:: orientation_vector_component_tolerance;
  form_tolerance: envelope_form_tolerance;
END_ENTITY;

ENTITY cylinder SUBTYPE OF (vt_face_surface);
  nom_location_vector: location_vector;
  nom_orientation_vector: orientation_vector;
  size : length_measure_with_unit;
  x_location_tolerance, y_location_tolerance: location_vector_component_tolerance;
  x_orientation_tolerance, y_orientation_tolerance:: orientation_vector_component_tolerance;
  r_tolerance: size_tolerance;
  form_tolerance: envelope_form_tolerance;
END_ENTITY;

2.3 Derived elements

The geometrical elements defining a coordinate system can be a face surface or elements derived from combinations of face surfaces. A derived element can be intersection point or line, line connecting two points, etc.

ENTITY derived_element
  ABSTRACT SUPERTYPE OF (ONEOF (intersection_point, symmetry_point, intersection_line, perpendicular_line, parallel_line, connection_line, symmetry_line, best_fit_line, perpendicular_plane, parallel_plane, connection_plane, symmetry_plane, best_fit_circle));
  basis : coordinate_system;
  name : STRING;
END_ENTITY;

ENTITY intersection_point
  SUBTYPE OF (derived_element);
  source_elements: SET[1:?] OF geometric_element;
  location: location_vector;
END_ENTITY;

ENTITY intersection_line
  SUBTYPE OF (derived_element);
  source_elements: SET[1:?] OF geometric_element;
  location: location_vector;
  orientation: orientation_vector;
END_ENTITY;

2.4 Coordinate systems

There can be several coordinate systems on a workpiece forming a tree structure of coordinate systems. The Workpiece Coordinate System (WCS) is the common basis. Position relations between surfaces with a special functional relationship, can be described in a Sub Coordinate System (SCS). Both these types of coordinate systems are clearly defined by the surfaces of the workpiece. The primary direction of a coordinate system is defined by a unit vector, fixing two rotational DOF. Second, the secondary direction is defined by a unit vector which are perpendicular to the primary direction. The secondary direction can be defined by the projection of an orientation vector not parallel to the primary direction. The third direction of the coordinate system is given by the cross product of the primary and secondary direction vectors. Finally is the origin of the coordinate system defined by a point, thereby fixing the three translation DOF.

ENTITY datum_feature_relationship;
  name: STRING;
  relating_element : geometric_element;
END_ENTITY;

TYPE geometric_element = SELECT
  (vt_face_surface, derived_element);
END_TYPE;
The Root Coordinate System (RCS) is a superior coordinate system defining the position of the whole workpiece in a machine tool, measuring machine or in a the assembly with other workpieces. The WCS can be defined in this RCS.

2.5 Workpiece

A workpiece consists of a set of geometric elements of planes, cylinders, lines, points and so on. These geometric elements are the surfaces as well as derived elements.

3 VECTORIAL TOLERANCES

3.1 Tolerances on location and orientation

The Vectorial tolerances are the tolerances on the location and orientation vector for the toleranced surface. The tolerance is maximum and minimum allowed components of the deviation vector. If the surface has any open DOF, the corresponding deviation vector components are zero. These components have therefore no tolerance.

If the orientation vector of the surface is \([0, 0, 1]\), orientation tolerances will be the tolerances on the \(E_x\) and \(E_y\) components of the orientation deviation vector. The orientation tolerances are the tolerances on the components perpendicular to the orientation vector. The location
tolerances will be the tolerances on the $x_0, y_0,$ and $z_0$ components of the location deviation vector. Figure 3 shows the tolerances of a plane. The translation DOF in $x_0$ and $y_0$ direction and the rotation around $z$-axis are open DOF, there are therefore no tolerances on these components. In the case of a cylinder, $E_z$ orientation and $z_0$ location is open and has no tolerance.

Figure 3. Vectorial tolerances on a plane

ENTITY location_vector_component_tolerance;  
name : STRING;  
tcs: tolerance_coordinate_system;  
max, min : length_measure_with_unit;  
length_measure_with_unit;  
END_ENTITY;

ENTITY orientation_vector_component_tolerance;  
name : STRING;  
tcs: tolerance_coordinate_system;  
max, min : REAL;  
END_ENTITY;

3.2 Tolerance Coordinate System, TCS

The TCS is the coordinate system where a tolerance of a surface is described. The TCS is always defined according to nominal geometry and is defined by the directions that is to be tolerated, i.e., defined by the function of the surface. The reference coordinate system to the TCS is the coordinate system where the tolerances are referring to. A TCS describes the tolerance space and the transformation of a deviation vector from the reference coordinate system to the tolerancing space for evaluation. For example: defining tolerances to a plane referenced but not aligned to the WCS, a TCS with the $z$-axis aligned with the plane normal can be introduced. When measuring the real surface, the vectors for this plane must be transformed from the WCS to the TCS for calculation of the deviation vectors and a evaluation of tolerance conformance.

ENTITY tolerance_coordinate_system;  
name : STRING;  
reference_coordinate_system : coordinate_system;  
origin : location_vector;  
x_axis, y_axis, z_axis : orientation_vector;  
END_ENTITY;

3.3 Size tolerances

The tolerances to the location and orientation are the tolerances to the position of the surface in WCS. Sizes are constant parameters which describe the size of the surface. A size is independent from the position of the surface, and needs therefore no basis in a coordinate System. The size tolerances will be maximum and minimum allowed deviation of the nominal sizes.
3.4 Tolerances on form deviations

In Vectorial tolerancing, form deviations are filtered out from location and orientation through the Gaussian best fit algorithm. The form deviations are therefore treated separately. The simplest way to limit the form deviations is to define an envelope tolerance region. This means the distance from any point on real surface to the best fit substitute surface must be within the tolerance limit.

Another option which is feasible for planes and cylinders, is to use tangential surfaces to express the form deviations. When tangential surfaces are used, the form deviations are described with the location and orientation or size deviation of one or two tangential surfaces relatively to the best fit surface. An advantage with the tangential surface approach is that this approach ensures 100% compatibility with ISO tolerance standards. Wirtz (1993) writes, however, that the difference between tangential and parallel region to best fit is in practice much smaller than the measuring uncertainty caused by low number of measuring points and the difference in measuring direction.

4 EXAMPLE

Here follows an example of the vectorial geometry and tolerances on a simple workpiece using the ISO standard for clear text encoding of exchange structure: ISO 10303-21 (STEP part 21). The workpiece, shown in Figure 4 and Figure 5, consists of seven surfaces, 6 planes and one cylinder. The function of the workpiece requires tight tolerances on position, perpendicularity to plane BP6, and cylindricity of the cylinder C1. Other important tolerances are the parallelity of planes P5 and BP6 and position of plane P5, as well as perpendicularity and flatness of the planes BP1, BP4 and BP6. Figure 5 shows corresponding tolerances on the workpiece according to the ISO 1101 standard. The planes BP6, BP1 and BP4 is here the datum planes A, B, and C which forms the datum system ABC similar to the WCS in the VT case. One important difference between VT and ISO are, however, VT planes are based on a Gaussian best fit substitution plane on the real workpiece. ISO planes on the other hand, are based on minimum movement contact planes. When coordinate measuring machines are used the VT surfaces is based on a all measured points, while a ISO interpreted contact surface is based on only few peak points. The latter requires a large number of measuring points on the surface. Other differences between ISO and VT are that ISO position tolerance includes form and orientation deviation, ISO orientation tolerance includes form deviation. The magnitude of each deviation type is not known. Furthermore, the direction of the orientation deviation is unknown.
Two TCS are introduced, TCS_root and TCS_wcs. TCS_root has the RCS as reference, the 6 components defining the WCS are toleranced here, and are thereby the tolerances of the WCS in the RCS. All other tolerances are defined in the TCS_wcs. Plane BP6 defines the primary direction (z-axis) of the WCS with the orientation vector of BP6. In other words; the unit vector \( \mathbf{k} \) of the WCS is identical with the orientation vector of BP6, on nominal as well as manufactured workpieces. The orientation of BP6 has therefore by definition no deviation in WCS. BP6 will, however, have deviation in the RCS, and the tolerances on BP6 must refer to the RCS. Since the orientation vector of plane A is identical with the primary direction of the WCS, these tolerances will be the tolerances on the primary direction of WCS in RCS. The secondary direction is defined by the intersection line between BP6 and BP1. This means Plane BP1 defines the secondary direction of the WCS with its \( \mathbf{E}_x \)-orientation component, and the \( \mathbf{E}_z \)-orientation component of BP1 is therefore toleranced in TCS_root referring to the RCS. The origin of the WCS is the intersection of plane BP6, BP1 and BP4. This means the location tolerance of BP6, BP1 and BP4 are toleranced in the TCS_root as well. This sums up 6 tolerance components of the planes defining the WCS, consequently 6 DOF. These components are defining the WCS, and the tolerances on them will be the tolerances of the WCS in the RCS. The remaining components; \( \mathbf{E}_y \)-orientation for BP1 and both orientations for BP4, as well as all other surfaces will be toleranced in the TCS_wcs.

ISO-10303-21; DATA;

```
#1 = LENGTH_MEASURE_WITH_UNIT('mm',0); /* 0 mm */
#2 = LENGTH_MEASURE_WITH_UNIT('mm',50); /* 50 mm */
#3 = LENGTH_MEASURE_WITH_UNIT('mm',25); /* 25 mm */
#4 = LENGTH_MEASURE_WITH_UNIT('mm',-20); /* -20 mm */
#10 = LOCATION_VECTOR('root_origin',#1,#1,#1); /* origin of rcs */
#11 = ORIENTATION_VECTOR('root_x',1,0,0); /* x-axis of rcs */
#12 = ORIENTATION_VECTOR('root_y',0,1,0); /* y-axis of rcs */
#13 = ORIENTATION_VECTOR('root_z',0,0,1); /* z-axis of rcs */
#14 = ROOT_COORDINATE_SYSTEM('root',#10,#11,#12,#13); /* rcs */
#15 = TOLERANCE_COORDINATE_SYSTEM('tcs_root',#14,#10,#11,#12,#13);
#30 = ORIGIN_DATUM('WCS_origin',#1,#1,#1); /* origin datum for WCS */
#31 = PRIMARY_DIRECTION_DATUM('WCS_x',0,0,1); /* primary direction for WCS */
#32 = SECONDARY_DIRECTION_DATUM('WCS_y',1,0,0); /* secondary direction for WCS */
#33 = WORKPIECE_COORDINATE_SYSTEM('WCS', #14,#30,#31,#32); /* The WCS */
```
#34 = LOCATION_VECTOR('origin',#1, #1, #1); /* Origin of tcs_wcs */
#35 = ORIENTATION_VECTOR('X',1.0,0.0); /* X-axis of tcs_wcs */
#36 = ORIENTATION_VECTOR('Y',0.0,1.0); /* Y-axis of tcs_wcs */
#37 = ORIENTATION_VECTOR('Z',0.0,0.1); /* Z-axis of tcs_wcs */
#38 = TOLERANCE_COORDINATE_SYSTEM('tcs_wcs',#33,#34,#35,#36,#37); /* tcs with wcs as reference */
#40 = LOCATION_VECTOR('lpb1',#1, #1, #1); /* location vector of plane bp1 */
#43 = LOCATION_VECTOR('lpb4',#1, #1, #1); /* location vector of plane bp4 */
#44 = LOCATION_VECTOR('lpb5',#1, #1, #1); /* location vector of plane bp5 */
#45 = LOCATION_VECTOR('lpb6',#1, #1, #1); /* location vector of plane bp6 */
#46 = LOCATION_VECTOR('lc1',#3, #3, #1); /* location vector of cylinder c1 */
#50 = ORIENTATION_VECTOR('obp1',0,-1.0,0); /* orientation vector of plane bp1 */
#53 = ORIENTATION_VECTOR('obp4',-1.0,0,0); /* orientation vector of plane bp4 */
#54 = ORIENTATION_VECTOR('obp5',0,0,1); /* orientation vector of plane bp5 */
#55 = ORIENTATION_VECTOR('obp6',0,0,1); /* orientation vector of plane bp6 */
#56 = ORIENTATION_VECTOR('oc1',0,0,1); /* orientation vector of cylinder c1 */
#60 = LENGTH_MEASURE_WITH_UNIT('mm',0.010); /* max */
#61 = LENGTH_MEASURE_WITH_UNIT('mm',-0.010); /* min */
#70 = LOCATION_VECTOR_COMPONENT_TOLERANCE('loctol_bp1',#15,#60,#61); /* location tolerance for bp1. Note that the TCS is the tcs with RCS as reference, this is because bp1 is used to define WCS origin. The 6 components used to define the WSC. These components therefore have no tolerances in the WCS, but in the RCS. This can also be viewed as the tolerances of the WCS in its superior coordinate system, RCS. */
#73 = LOCATION_VECTOR_COMPONENT_TOLERANCE('loctol_bp4',#15,#60,#61);
#74 = LOCATION_VECTOR_COMPONENT_TOLERANCE('loctol_bp5',#38,#60,#61);
#75 = LOCATION_VECTOR_COMPONENT_TOLERANCE('loctol_bp6',#15,#60,#61);
#76 = LOCATION_VECTOR_COMPONENT_TOLERANCE('loctol_x_c1',#38,#60,#61);
#77 = LOCATION_VECTOR_COMPONENT_TOLERANCE('loctol_y_c1',#38,#60,#61);
#78 = LOCATION_VECTOR_COMPONENT_TOLERANCE('loctol_z_c1',#38,#60,#61);
#79 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_x_bp1',#38,0.001,-0.001);
#80 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_y_bp1',#38,0.001,-0.001);
#81 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_z_bp1',#38,0.001,-0.001);
#82 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_x_bp4',#38,0.001,-0.001);
#83 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_y_bp4',#38,0.001,-0.001);
#84 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_z_bp4',#38,0.001,-0.001);
#85 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_x_bp5',#38,0.001,-0.001);
#86 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_y_bp5',#38,0.001,-0.001);
#87 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_z_bp5',#38,0.001,-0.001);
#88 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_x_bp6',#38,0.001,-0.001);
#89 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_y_bp6',#38,0.001,-0.001);
#90 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_z_bp6',#38,0.001,-0.001);
#91 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_x_c1',#38,0.002,-0.002);
#92 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_y_c1',#38,0.002,-0.002);
#93 = ORIENTATION_VECTOR_COMPONENT_TOLERANCE('oritol_z_c1',#38,0.002,-0.002);
#94 = LENGTH_MEASURE_WITH_UNIT('mm',0.010); /* max form deviation */
#95 = ENVELOPE_TOLERANCE('form tolerance for all surfaces',#94);
#96 = LENGTH_MEASURE_WITH_UNIT('mm',0.050); /* max size deviation */
#97 = LENGTH_MEASURE_WITH_UNIT('mm',-0.050); /* min size deviation */
#98 = LENGTH_MEASURE_WITH_UNIT('mm',10.000); /* radius of cylinder c1 */
#99 = SIZE_TOLERANCE('size tolerance of c1',#96,#97); /* Size tolerance of cylinder c1 */
#100 = PLANE('BP1',#33,#40,#50,#70,#80,#81,#95); /* Plane bp1 */
#103 = PLANE('BP4',#33,#43,#53,#73,#87,#95); /* Plane bp4 */
#104 = PLANE('BP5',#33,#44,#54,#74,#88,#95); /* Plane bp5 */
#108 = PLANE('BP6',#33,#45,#55,#75,#90,#95); /* Plane bp6 */
#109 = CYLINDER('C1',#33,#46,#56,#76,#92,#99,#95); /* Cylinder c1 */
#110 = LOCATION_VECTOR('loc_point1',#1, #1, #1); /* location of intersection point */
#111 = LOCATION_VECTOR('loc_line1',#1, #1, #1); /* location of intersection line */
#112 = ORIENTATION_VECTOR('ori_line1',#1, #1, #1); /* orientation of intersection line */
#113 = INTERSECTION_POINT('point1',#100,#103,#105,#110);
#114 = INTERSECTION_LINE('line1',#100,#105,#111,#112);
#115 = DATUM_FEATURE_RELATIONSHIP('origin_ref',#113,#30); /* Origin is defined by the intersection point between BP1, BP4 and BP6 */
#116 = DATUM_FEATURE_RELATIONSHIP('pr_dir_ref',#105,#31); /* Plane BP6 defines the primary direction of the WCS */
#117 = DATUM_FEATURE_RELATIONSHIP('sec_dir_ref',#114,#32); /* The secondary direction is defined by the intersection line between BP6 and BP1 */
#200 = WORKPIECE('Square plate with hole', #33, #100, #101, #102, #103, #104, #105, #106); ENDSEC; END-ISO-10303-21;
5 CONCLUSIONS

This paper shows the formal description of VT with examples using the EXPRESS language developed by STEP. Obtained result can be used in general to help understand how to use VT. The description includes the procedures of the definition of coordinate systems, coordinate system tree with root coordinate system, workpiece coordinate system and sub-coordinate system, as well as the relations between tolerances, tolerance coordinate systems and the tolerances on surfaces defining a coordinate system.

In the future work, the obtained EXPRESS definition will be a basis for a formal Comparison of VT and other tolerancing methods, such as tolerances after the ISO standards and TTRS. ISO/DIS 10303-47 of Integrated generic resources: Shape variation tolerances is to provide a general description of dimensioning and tolerancing. This paper will be an example to show how the general description can be applied. The obtained EXPRESS definition will be used as a reference description of product data model based on Vectorial geometry and tolerancing.

6 REFERENCES

ISO 10303-11, 1995, Product data representation and exchange - Description methods: The EXPRESS language manual
ISO/DIS 10303-47, 1995, Product data representation and exchange - Integrated generic resources: Shape variation tolerances

*Vectorial dimensioning and tolerancing*. Proposal to new work item to ISO TC 10, ISO TC 3
Wirtz A., Gaechter C., Monstein B., Putzi C. (1990), Die Vektorielle Tolerierung, ein werkzeug zur Funktionsanalyse, *Technische Rundschau (Switzerland)*, nr. 41/90, Hallwag Verlag, Bern

7 BIOGRAPHY

Kristian Martinsen is dr.ing. from the University of Trondheim, Department of production and quality engineering. He is currently working as a STA Post-doctoral fellow at Mechanical Engineering Laboratory. His interests are tolerancing methods, coordinate measuring and accuracy and control of machine tools.
Dr. Toshio Kojima is a chief senior researcher of Mechanical Engineering Laboratory, AIST, MITI and also a visiting professor, the graduate school, Science University of Tokyo. He is currently interested in the formulation and the implementation of manufacturing process and product data using algebraic method.
PART SIX

Shop Floor Systems
Abstract
This paper describes the components and functionalities of a Modelling Workbench (MW) to be used within the context of the so-called Purdue Enterprise Reference Architecture (PERA) methodology. The different life-cycle phases of an enterprise integration project and the model building blocks provided are presented and described. A short description of the proposed pre-standard ENV 40 003 basic reference model, which provides the context for the so-called executable models, is first presented. This is followed by an explanation of the use of the MW within the context of the Purdue Enterprise Reference Architecture life-cycle support. While most of the issues discussed in this paper are confined to the shop floor environment and related applications, the general underlying concepts are relevant to manufacturing systems in general.

Keywords
Enterprise Reference Architectures, Executable Models, Life-cycle, Modelling, Enterprise Integration

1. INTRODUCTION
The research work presented in this paper was undertaken in parallel with and drawing from work done in the course of two ESPRIT projects (EP-5478 Shop-Control and EP-8865 Real-I-CIM), and it mainly concerns the construction of a Modeling Workbench (MW) supporting different activities within the context of enterprise integration projects. This MW provides the systems integrator with a set of “Lego” type components that can be used to model the actual manufacturing floor under different interacting perspectives of views (e.g. information systems, manufacturing equipment and company organization). Simulation runs may then
follow this modeling activity, allowing alternative design solutions to be evaluated and compared. Further stages in the integration project life-cycle, such as component stepwise testing and integration or components replacement for system upgrade are also supported [Schulte, Ferreira, Soares].

Regardless of the need for the above described support to modeling & simulation activities which are typical of a consultancy project, a most important issue remains open, and that is the actual "usability level" of the MW for each phase of the integration project life-cycle. This usability threshold is directly related with the abstraction level required from the user while using any type of tool, and if the correct balance is not achieved it may happen that the best tool may remain unused.

While the research work undertaken falls within the broad scope of enterprise modelling, the MW was developed with the aim of supporting the complete shop floor applications life-cycle. The MW will be presented in the context of the Purdue Enterprise Reference Architecture (PERA) [Williams], highlighting the different life-cycle phases and the model building blocks provided. This explanation starts by introducing the concept of executable model, with a short description of the proposed pre-standard ENV 40 003 basic reference model [CEN]. This is followed by a presentation of the MW in the context of the PERA life-cycle support to enterprise integration activities. The discussion of relevant issues that arise when building a usable tool set to support the enterprise integration life-cycle closes this paper.

The issues tackled here are confined to the scope of the so-called shop floor control or manufacturing execution systems level, and to the corresponding functions / applications. The underlying concepts used in the course of the explanation can be nevertheless of general use in manufacturing.

2. FRAMEWORK FOR ENTERPRISE MODELING [CEN]

The pre-standard ENV 40 003/1990 CIM Systems Architecture - Framework for Enterprise Modeling, comprises three dimensions which cover the concepts needed for enterprise modeling: the first is concerned with the development and evolution of the model, starting from a statement of the requirements to a processable or executable model, this dimension being the Model of an enterprise; the second is concerned with the structure and behaviour of a model which considers appropriate aspects of an enterprise, this being the dimension of View; the third dimension is concerned with the degree of particularization which identifies the set of possible models, this is the dimension of Genericity. Whereas the last two dimensions are related with the actual modelling views and modelling strategy, the first one is inherently related with the actual model life cycle. This means that the Model or the Model components derived from the ENV 40 003 should ultimately be computer executable thereby enabling the daily execution of enterprise tasks.

The Enterprise Model Execution and Integration Services (EMEIS in figure 1) appear, at the most general level, as a set of services allowing the interpretation of a model or model components for the operation of Manufacturing Technology Components (entities required to carry out the manufacturing processes, i.e. physical operations). Both this model or model components were previously developed in a Model Development Environment, which makes use of the Model Development Services, (MDS).

The EMEIS include General IT Services, these being used as a platform upon which specific services are built. These services deal with the functionalities required for the use of a
model or functionalities that are specific to manufacturing or enterprise systems integration. The boundary between MXS and General IT Services is not rigidly defined, the application programs are components of the EMEIS and may provide a service or just be a user of services.

It is likely that, seen from the viewpoint of a user, vendor or system integrator, a particular EMEIS will comprise both General IT services and enterprise integration-specific services, these having been derived from General IT capabilities available and from enterprise integration-specific requirements.

![Figure 1 - ENV 40 003 basic reference model.](image)

This very simplified representation of EMEIS and its IT support can be refined into three basic components (please refer to figures 1 and 2) which are now described. The Model Execution Services (MXS) embed services which transform a model component into an executable application entity. These services interpret (or instantiate) the model component as a "model-based executable entity", thus converting it into a "runnable" application. These services are dependent on the modeling technique used. The MXS operation services are built by all the particular IT services that an integration project would require above existing General IT services. General IT services are obviously independent from the modelling technique used. The General IT Services are also not dependent on the integration project. These services can be characterised as systems that provide: portability of applications, internetworking between heterogeneous systems, and distribution transparency where processing is distributed.

3. THE MODELING WORKBENCH, A MODELING ENVIRONMENT FOR THE LIFE-CYCLE SUPPORT

3.1 Introduction

The best use of the life-cycle concept implies that results obtained for each project phase should be adequately reused in the immediately following phase(s) [Williams]. However, this objective is not easy to accomplish as the different tools used in the various life-cycle phases,
This section will show how this problem was tackled by using an integrated modelling workbench (MW) in the context of the PERA methodology. This MW is based on an off-the-shelf object-oriented material flow simulation tool [Simple++]. This tool has been further enhanced in the course of previous work to incorporate Information Systems Architecture modelling as well as techno-organizational modelling capabilities [Ferreira, Martins, Soares]. Whereas in the former case the user is provided with the means to formally describe and simulate the actual software system and architecture, in the latter case the user is able to model the human & organizational aspects of the interaction with the shop floor information system. An integrated tool set covering these three different but complementary aspects in manufacturing, i.e. machines, computers and people, was therefore achieved.


Figure 2 - The Modeling Workbench in the Context of the PERA
Figure 2 portrays the different phases and activities encompassed by the Purdue Enterprise Reference Architecture (PERA) [Williams], as well as the proposed use of the available integrated tool set for the life-cycle support to enterprise integration projects [Soares, Martins, Ferreira]. As illustrated, the work undertaken during the requirements analysis phase produces a document set containing both the (manufacturing) operations and the (management/ control) information functional networks. As proposed by PERA, this network structure information is then used in the design of the three implementation architectures, i.e. information systems architecture (ISA), human and organizational architecture (H&OA) and manufacturing equipment architecture (MEA), which are shown in figure 3.

3.2 Design Phase

At this point of the design phase, the integrated modelling environment available provides the user with the adequate means to model the Information System Architecture (ISA). The first step is to use a DFD-based CASE tool [TeamWork] to build an agent-based description [Martins]. During this phase, the designer is also provided with a set of building blocks to support the MEA [Simple++] as well as the H&OA design [Soares]. As shown in figure 2, the ISA agent-based model built using the CASE tool is then imported into the MW as an SDL (Specification and Description Language) model [Belina]. An integrated model reflecting the three implementation architectures, as well as their interfaces and interactions is therefore obtained. This model may then be tested, verified and evaluated within the modelling and simulation environment provided by the MW (figure 2 and 3).
3.3 Implementation Phase

In the context of the ENV 40 003, these models must now be converted into their executable version, so that they may be able to run on top of the Model Execution Services (MXS in figure 1). The next phase, implementation, reuses to their maximum extent the models built during the previous design phase, thereby providing three complementary results: interacting software models of information management and control system components (ISA); activities reflecting human with the information system (H&OA); shop floor resources layout and material flow control models (MEA).

During this phase, the ISA agent-based SDL specification is exported into an SDL case tool, which provides all the means for generation of executable code. On the other hand, the previously built MEA material flow simulation control model is released for execution and can be reused to control the everyday shop floor operations. Finally, the results of the techno-organizational evaluation work should provide some insight on crucial and increasingly acknowledged organizational issues, i.e. the third implementation architecture built of organizational units (e.g. cells, lines, working group, etc.) as well as on their interactions.

3.4 Build and Operation Phases

Figure 4 focus on the two last life-cycle phases. On top of a general purpose infrastructure providing distributed communication facilities, a set of Manufacturing Execution Services (MXS) is available to facilitate the model release for operation.

Whereas software application models defined for the ISA may be used to generate the actual executable C/C++ code, the MEA control model which is used in the actual control of everyday manufacturing operations runs on top of the Simple++ simulator (Sim in figures 3 and 4). The MEA model is also used for a graphical animated shop floor monitoring display.
4. THE MODELLING WORKBENCH USERS

Along the PERA life-cycle the end-user has a most special role. However, when referring to enterprise model users, four types of people are involved: the modellers, the analysts/consultants, the managers (or decision makers) and the operators, while keeping in mind that an individual user may belong to more than one of these types [Fraser]:

- modellers are usually skilled in methods that originate in science, engineering or information technology;
- managers make business decisions on the basis of (implicit or explicit) models and their analysis, based on their analysis expertise;
- finally, the operators carry out activities which are the results of business decisions, and they might require to visualize parts of the model to generate a set of activities needed for themselves or to execute their work.

The end-user relevance in the modelling life-cycle is usually disregarded, a possible reason for this fact being his lack of ability in the use of modelling languages. This problem should consequently be addressed, since the use of an adequate modelling language will grant both the analyst and the modeller the possibility of making the best use of the end-user expertise in its often very specialised domain.

In this context, one of the current concerns is to encourage the end-user involvement in the early stages of the modelling process by providing adequate modelling languages. The use of these languages combined with industry specific reference models should both foster the user involvement in the modelling process and significantly reduce the time needed for the model construction.

5. CONCLUSIONS

This paper has presented a contribution to solving technological problems raised by enterprise integration projects. The integration of different off-the-shelf tools supports projects right from the analysis and design stage. A very important problem is yet to be solved, this being the interaction with the end-user. In the design and in the subsequent phases, the end-user interaction with the involved concepts may be achieved through demonstrations within the MW simulation environment. In the requirements analysis phase this interaction is however not yet covered, and becomes more critical since problem understanding needs to be achieved and documented through the use of a common modelling language.

6. REFERENCES


Camarinha -Matos and Hemideh Afsarmanesh, Published by Chapman & Hall, Lisboa, June 96.


7. BIOGRAPHY

João José Pinto Ferreira was born in 1964. He holds a degree in Electric Engineering by the University of Porto (1987) as well as MSc degree by the University of Porto (1991) and a PhD degree by University of Porto (1995).

From 1987 he has been an Assistant Lecturer at the Faulty of Engineering University of Porto. Since 1995, as an Assistant Professor, he is responsible for the chair “Architectures for Enterprise Integration” at the Masters Course at the Engineering Faculty of the University of Porto.

Since 1987 he is also a researcher at INESC (Instituto de Engenharia de Sistemas e de Computadores) in Porto, at the Manufacturing Systems Engineering Group. He has been involved in several national and european R&D projects, most notably Esprit projects 5478-SHOP-CONTROL and 8865-REAL-I-CIM.

José Manuel Mendonça holds a degree in Electric Engineering by the University of Porto (1977) and received his PhD degree by Imperial College, London (1986).

His professional activities include Brown Boveri & C.ie, Switzerland (1977), Denmark's Technical Highscholl (1978/79) and EDP - Portuguese Power Corporation (1979/80).

From 1980 he has a teaching position at the Faulty of Engineering University of Porto, being responsible for chairs in Quality Management, Computer Control and Manufacturing Systems Integration.

From 1987 to 1996 he has been a senior researcher at INESC (Instituto de Engenharia de Sistemas e de Computadores) in Porto, where he was the head of the Manufacturing Systems Engineering Group. He has been responsible for numerous national and European R&D projects, most notably Esprit projects 5478- SHOP-CONTROL and 8865 REAL-I-CIM.
Multiprotocol Ethernets on the Shop-Floor

J. Pütter
Chalmers University of Technology
Department of Production Engineering
S–412 96 Gothenburg, Sweden
Phone: +46 31–7721291, Fax: +46 31–7723819
E-mail: puetter@pe.chalmers.se

Abstract
Computer networks are indispensable for any CIM implementation, since a computer network is the only transport mechanism available for sharing computer-based information. The rapidly developing computer and IT fields has given manufacturing industry new opportunities to increase its competitive strength. It can make efficient use of new developments by adapting itself to widely spread standards and de facto standards. This article aims to show that the popular Ethernet technology can be successfully used as a multiprotocol CIM network on the shop-floor. Examples of Ethernet-based networking practices from three different European manufacturing companies are given.

Keywords
Computer networks, communication protocols, computer-integrated manufacturing, Ethernet

1 INTRODUCTION

1.1 CIM and networking

The basis for an enterprise-wide CIM (computer-integrated manufacturing) concept is the integration of various functions and technologies. One of the key factors to compete effectively in manufacturing is operational flexibility. Integration by linking computer-based tools over a common network has been widely accepted as one of the main ways to achieve this flexibility. ‘The key to CIM is systems integration and an incremental CIM approach’, according to (Ahmadi et al., 1991).
In general, computer networks connect different computer types, running different operating systems and using different network protocols – a distributed, heterogeneous, networked, multivendor computing environment. Such an environment is utilized to share:
1. Data stored in files or databases.
2. Resources such as printers, backup devices, modems, WAN (wide-area network) links, etc.
3. Programs where users should be able to run all software tools required to get their job done efficiently – even if some programs need a computer platform different from what the user has on his desktop. This can be accomplished in two completely different ways (Pütter, 1995):
   a. Through emulation: e.g. the software SoftPC emulates a personal computer (PC) running MS-DOS (Microsoft disk operating system) on a Macintosh;
   b. Over the network: Remote control, e.g. telnet, rlogin, X-Windows, etc.

A literature review on CIM networking problems (Snyder, 1991) found that ‘the networking solution for CIM may be considered as a part of the total organization’s establishment of a rational and effective network architecture’. Snyder categorized the problems he found into:
- multiple vendor installations;
- the scarcity of off-the-shelf software packages for CIM;
- immaturity of connectivity products;
- network management.

A survey (Penning, 1994) conducted in mid-1993 asked 164 small (less than 200 employees) U.S. manufacturing enterprises about, among other things, their current networking status. Penning found that nearly all of the 84 respondents with on-site networks reported that they were used for office and administrative functions. Networks for shop-floor applications were operative in 43 of the 84 enterprises with networks installed. About two-thirds of those familiar with their network topology were using Ethernet. Computer platforms used were PCs (87 respondents), workstations (31 respondents), minicomputers (17 respondents), dedicated systems (10 respondents), and mainframes (6 respondents). This indicates that computer environments are often heterogeneous, a conclusion also found in (Ahmadi et al., 1991) and (Snyder, 1991).

1.2 Shop-floor networks and their protocols

A modern shop-floor network is utilized much the same way as office networks, but three additional communication aspects have to be considered:
1. Electromagnetic interference (EMI) is often higher due to the presence of high-current power lines, electric motors, arc welders, arc furnaces, motor generators, etc., which can disrupt network communications.
2. Real-time requirements are common, the network must be able to successfully deliver packets to their destination under a deterministic time-frame.
3. Computer-to-computer communication without human intervention, e.g. automated transactions between distributed control systems in a manufacturing cell.

A designer of shop-floor networks should take these aspects into consideration.
In the early eighties, General Motors stated that 30% to 50% of their automation budget was spent on building custom interfaces between incompatible factory systems. This led to the development of MAP, the manufacturing automation protocol. The use of a standard communications protocol, well suited for shop-floor automation, should obviate the need for expensive customized interfaces and thereby significantly reduce costs and lead times. However, MAP is still very expensive, it is not widely spread, open systems interconnection (OSI) communications are very complex, and a competitive MAP market, so far, has not evolved. Also, the MAP 3.0 specification initially only allowed two cable types: the expensive IEEE 802.4 Token Bus broadband and the uncommon 5 Mbps IEEE 802.4 Token Bus carrierband.

In 1992, the IEEE 802.3 CSMA/CD (carrier sense multiple access with collision detection) networks were included in the MAP standard. This should pave the way for a broader acceptance of MAP as a shop-floor protocol. It should be noted that the IEEE 802.3 standard and Ethernet, being almost identical, are incompatible due to a difference in the two-byte type/length field in their respective frame formats. Both 802.3 and Ethernet can be used to transport the MAP protocol; it is a question of how the network interface board is set up. Also, a few different Ethernet versions have been presented since 1980. This paper deals exclusively with the latest version often referred to as ‘Ethernet’ or more specifically, DIX Ethernet version 2.0 (Digital-Intel-Xerox).

As Ethernet became part of the MAP standard, it resolved several pending issues:
1. The network infrastructure can be more easily and inexpensively implemented by using popular and readily available Ethernet technology.
2. The knowledge and experience with Ethernet-based networks is far more extensive than it is with any other network type.
3. Connectivity to Ethernets is simpler and cheaper than it is to Token Bus networks. The CSMA/CD media-access method used in Ethernet has one well-known drawback – the network can get congested, resulting in increasing response times, when subjected to high loads under certain conditions (Boggs et al., 1988). The end effect is that real-time demands cannot be guaranteed in all situations. A family of networks, collectively called fieldbus, has found a market in interconnecting sensors, actuators, or to distribute I/O points in relatively limited areas. A fieldbus is inexpensive, has good real-time performance, is uncomplicated, and can cover a manufacturing cell. Popular fieldbuses are Profibus, CAN bus, Interbus-S, and FIP. A fieldbus typically is limited to 100 meters, 30 nodes, small packets (a few bytes), and 1 Mbps of bandwidth.

A closer analysis of the MAP protocol suite reveals that most application-layer functions can be successfully replaced with other, more widely used, protocols. There is one important exception: the manufacturing message specification – MMS or ISO 9506. The MMS protocol is probably the most interesting application-layer OSI protocol to the manufacturing environment. This extensive protocol is designed for the remote control and monitoring of industrial devices such as CNC (computerized numerical control) machines, robots, PLCs (programmable logical controllers), AGVs (automated guided vehicle), cell controllers, etc. It is an internationally standardized messaging system for exchanging real-time data and supervisory control information between networked devices. The messaging services provided by MMS are generic enough to be appropriate for a wide variety of devices, applications, and industries. There are no alternatives to MMS today, at least no standardized ones. MMS would facilitate shop-floor integration considerably; therefore is it highly desirable that MMS gain wider use.
1.3 Multiprotocol Ethernets, Internet, and TCP/IP

In Europe, about 50% of all installed local-area networks (LAN) are based on Ethernet technology (Smythe, 1993). It is stated that in the U.S., Ethernet comprises over 61% of the installed LANs in use today (Adams, 1990). This market penetration has lead to widespread knowledge and experience with the technology and a highly competitive LAN market with a rich array of products. (Smythe, 1993) and (Shaw, 1989) mention six different integrated cable architectures (10BASE5, 10BASE2, 10BASE-F, 10BASE-T, 10BROAD36, and 1BASE5) as well as several vendors of network interface cards, terminal servers, repeaters, hub units, bridges, switches, and routers – the necessary components to build a flexible networking infrastructure for both office and shop-floor environments. Lately, the 100 Mbps Ethernet (100BASE-TX, 100BASE-F) is establishing itself on the market as the high-speed Ethernet alternative to the traditional 10 Mbps version. Also, IEEE currently is working on the GigaEthernet standard; products are expected to appear on the market during 1997.

Another important Ethernet quality is its ability to simultaneously accept several communication protocols. Most, if not all, popular LAN protocols can use Ethernet, for example:

1. TCP/IP (transmission control protocol/Internet protocol).
2. Apple’s EtherTalk.
4. NetBIOS (network basic input output system) and NetBEUI (NetBIOS extended user interface).
5. Digital’s DECnet and LAT (local area transport).

Different protocols can successfully exist side by side on an Ethernet cable without knowing about each other – interprotocol integration is not the case. Instead of using gateways to convert between protocols, one protocol can be selected as the integrating protocol – the network Esperanto (Pütter, 1995).

The global Internet has experienced a tremendous growth during the nineties; up to 50 million users have access to it, according to some estimates. The reason behind the rapid growth is Internet’s useful content and its ability to interconnect people – features advantageous to all companies, organizations, and individuals, just like the telephone system. An example of how the manufacturing industry can use Internet in the future is given in (Coyne et al., 1994).

This is a research project aimed at creating an ‘advanced collaborative open resource network (ACORN)’ – an infrastructure to create an electronic manufacturing community able to design and sell engineered products in competitive markets as well as conduct research and development by collaborating over a network.

Only TCP/IP is allowed on the Internet. The TCP/IP family of protocols can be used to transport data around the world as well as across the hallway. This made TCP/IP the most widely spread and used protocol available. Most, if not all, computers can use the protocol suite for its communications. Other reasons for wanting to use TCP/IP on the shop-floor are:

- TCP/IP is a widely used de facto standard. The specifications are publicly available, for free.
- TCP/IP is the only protocol used on Internet. Due to Internet’s popularity, TCP/IP will probably end up being used on the shop-floor anyway.
- Some 50 million users have knowledge and experience with it.
- It scales extremely well, an important flexibility issue – there is no risk for outgrowing it.
- MAP’s application-layer protocols can be successfully replaced by TCP/IP protocols, except for MMS, as mentioned earlier.
2 THREE INDUSTRIAL EXAMPLES

This section briefly describes computer networking practices in three different European manufacturing companies. Their common denominator is the successful use of multiprotocol Ethernet implementations.

2.1 A wheel-axle manufacturing plant

The Ethernet-based plant-wide office and shop-floor LAN connects more than 150 nodes of various types: approximately 100 PCs running MS-DOS, WfW (Windows for Workgroups) or OS/2, 40 UNIX and VMS workstations, 15 terminal servers, and several PLCs are connected to the network. The LAN carries TCP/IP, DECnet, LAT, OSI, NetBIOS, and NBT (NetBIOS-over-TCP/IP – Aggarwal et al., 1987a; Aggarwal et al., 1987b; Hunt, 1995) – a multiprotocol Ethernet in a heterogeneous computing environment. A router/bridge interconnects the LAN with two leased 64 kbps WAN lines. Only TCP/IP is routed; DECnet and LAT are bridged over the WAN. Also, a dial-up Internet connection is currently being evaluated.

The network has been used for several years, while continuously being extended and upgraded as new computers or manufacturing devices needed access to resources available over it. An analysis showed that this one-segment Ethernet violated three of the five Ethernet guidelines presented by (Boggs et al., 1988):

• the network used very long cables to cover all buildings;
• there were many hosts on the single segment;
• real-time nodes where not separated from bulk-transfer nodes.

Therefore, the network recently was split into seven logical segments by means of a high-performance, multiport Ethernet switch (i.e. a fast, multiport bridge). The primary objective was to decrease the network’s vulnerability to malfunctioning nodes. The secondary objective was to increase bandwidth by making more effective use of Ethernet technology. The introduction of the switch reconfigured the network into a collapsed backbone topology with bridged segments. This significantly shortened the length of each segment and decreased the number of hosts per segment. As a consequence of this reconfiguration (1) the network now better conforms with the guidelines presented by (Boggs et al., 1988), (2) available bandwidth has increased significantly because local network traffic will stay local, instead of flooding all nodes as it did before the switch was installed, and (3) the entire network is more flexible for future changes and expansions due to the modular switch design. Both points (1) and (2) above increase the network’s real-time characteristics.

There never was a problem with EMI radiation at this site as its EMI levels are considered low. Fiber-optic cables are used between buildings, being immune to EMI, while factory segments use shielded cables as a way to decrease the influence of EMI.

The network is used for sharing data, resources, and programs. Most PCs are used either as desktop computers or as cell controllers. Workstations are used as engineering workstations, MRP II (manufacturing resource planning) servers, AS/RS (automatic storage/retrieval system) controllers, and AGV controllers. The manufacturing equipment is connected to the network either through terminal servers or indirectly through cell controllers.
The WAN connection is used to convey production plans, e-mail, and product specifications, such as CAD (computer-aided design) files and assembly structures, to the headquarters some 300 km away.

An increasingly important part of the company’s core business relies on the network. Network downtime would quickly lead to halted production processes, resulting in costly logistic problems. Thus, reliability and flexibility are crucial.

2.2 An automotive body-shop

In 1994, a new body-shop line was taken into production. It is a highly automated and flexible, robot-based, spot-welding line that manufactures automobile bodies. More than 80 robots, 50 PLC systems, 25 PCs (MS-DOS, WfW, NT, and OS/2), 12 VMS and UNIX workstations, 100 terminals, 10 printers, 50 read/write units for an ID system, and 9 cameras are connected to the Ethernet-based factory LAN. The network used more than 40 km of copper and fiber cables interconnected with two multiprotocol routers and six hub units. Fiber cables are used throughout the shop-floor to eliminate the influence of EMI radiation since the line incorporates many EMI-generating spot-welders. Real-time demands are secured by separating bulk-transfer nodes from real-time nodes and by keeping segment length down. The Ethernet factory LAN carries TCP/IP, DECnet, LAT, 3270-emulation (DECnet to an SNA gateway), and MMS (OSI on Ethernet) – a multiprotocol Ethernet in a heterogeneous computer environment.

The factory LAN is connected to the company WAN through both routers for redundancy. The WAN has a connection to the Internet, the two being separated by a comprehensive firewall system to protect the LAN from intruders.

The network is used to share data, resources, and programs, such as production control, on-line geometric measurements, quality monitoring, historical production records, production monitoring, synchronization of real-time clocks, backups, off-line programming, e-mail, access to shop-floor software from development departments etc.

The network is a mission-critical component for the line to properly operate. Network downtime would, after less than one minute, lead to halted production processes, resulting in costly shortages of car bodies downstream.

2.3 A small electronics manufacturing company

For several years, this 40-employee company has been using an Apple LocalTalk network with about 8 Macintoshes, a file server, and a laser printer in the office environment. This was an office-only network; the factory PC initially did not need a network connection. A traditional serial-line terminal network also was used to access the minicomputer that ran MRP II and administrative software such as accounting, storage, purchasing, salaries, and sales. This setup, in terms of performance and flexibility, was not able to exploit major advances in computers and IT (information technology) that offer significantly improved performance and functionality. It is complicated to connect PCs and terminals to the LocalTalk, and the network was limited to the AppleTalk protocol. A major upgrading was necessary to maintain the company’s competitive strength.

Recently they installed a 40-node 10BASE-T Ethernet network to replace their now obsolete LocalTalk and serial-line networks and to extend the network into their factory. Special attention to protect the network from EMI influence was not needed since EMI levels are low, comparable
Multiprotocol Ethernets on the shop floor

341

to normal office levels. Real-time demands are not present. The new network hosts about 10 Macintoshes, 10 PCs (MS-DOS, WfW, Windows95, and OS/2), a laser printer, and a UNIX server. It carries TCP/IP, NBT, and EtherTalk protocols. Also, a dial-up Internet connection is currently being evaluated.

The network is utilized much the same way as before, to share data, resources, and programs across computer platforms. Now, however, it is a multiprotocol Ethernet in a heterogeneous computing environment. They successfully integrated this environment into a versatile and flexible system. Office and factory are integrated and several new technologies, e.g. CAD, CAT (computer-aided testing), CAE (computer-aided engineering), DNC (direct numerical control), CAM (computer-aided manufacturing), a simulation package, and shop-floor control, were easily added to the new network. The new system is flexible and has enough bandwidth for future growth. Their business process is relying heavily on the network – that has not changed.

3 DISCUSSION

The examples given above, each representing a different type of manufacturing industry, indicates that multiprotocol Ethernet-based CIM networks can be successfully implemented and used in shop-floor environments. Another common denominator is the use of modern client/server technology as found in many downsized (Baker, 1992) office environments. The literature reports similar setups (Ahmadi et al., 1991; Snyder, 1991; Baker, 1992; Bartlett et al., 1994; Casey, 1990), to mention but a few.

It is necessary to follow major trends in the computer and IT fields in order to utilize its rapid development. For the past few years, those trends include Ethernet, personal computers, workstations, UNIX, the Internet, downsizing, TCP/IP, LAN, and graphical user interfaces. The office and consumer markets are the main followers of these trends.

Routers, switches, bridges, repeaters, and nine media-options (six 10 Mbps and three 100 Mbps) make Ethernet very flexible. Gigabit Ethernet and new developments on VLAN technologies (virtual LAN) for switches and 'one-armed' routers promise new possibilities for the near future. The Ethernet technology can easily be designed to meet most demands on bandwidth and topology. An Ethernet has a nominal bandwidth of 10 or 100 Mbps. The normal network load, however, should be kept significantly lower, 10% of nominal bandwidth is normal procedure, as a margin for sporadic peak loads. When the load increases, the result will be more packet collisions, jams, and subsequent retransmissions. This can lead to a congested network under certain conditions (Boggs et al., 1988). This is a well-known aspect of the CSMA/CD media-access protocol used in Ethernet. By closely following the design guidelines presented by (Boggs et al., 1988), congestion can effectively be avoided and real-time performance will not suffer. However, it is often advantageous to use a fieldbus-based network on the lowest levels of shop-floor automation. Higher automation levels normally implies lower real-time demands, higher bandwidth needs, and more sophisticated protocols. This is where an Ethernet is better suited than a fieldbus network.

Although an Ethernet easily carries most common LAN protocols, it does not integrate disparate protocols. However, all devices connected to the same Ethernet and using the same communications protocol do understand each other; this can be described as islands of integration. The interprotocol integration has to be made elsewhere, by gateways or by selecting one protocol to be the Esperanto – the integrating protocol (Pütter, 1995).
Noise, such as EMI, is a threat to all electronic devices and they are all exposed to it to various degrees. The noise induced by EMI onto the network can affect data integrity. Error detection and retransmission features in the protocols will guarantee error-free data transmissions, but at the cost of additional network traffic. High noise levels would lead to high retransmission rates, degrading network performance. (Adams, 1988) conducted a comprehensive study on the correlation between EMI and Ethernet performance. The EMI levels he measured during several manufacturing site visits, turned out to be at least one order of magnitude lower than the Ethernet specification allows. Later laboratory tests, he reports, showed that Ethernet cables were unaffected up to a field strength twenty times the specification for 10BASE2. The 10BASE5 specification tolerates even higher levels. (Casey, 1990) states that 10BASE-T is not considered suitable on the factory floor. The reason is that a properly installed coaxial cable has better noise immunity at Ethernet frequencies than does an unshielded, twisted-pair cable. Fiber cables are immune to EMI, making their use a suitable strategy in harsh environments.

The three industrial examples presented above experienced some of the four main CIM networking problems reported by (Snyder, 1991):

- **multiple-vendor installations**: were a problem mostly with older equipment;
- **the scarcity of off-the-shelf software packages for CIM**: was not investigated;
- **immaturity of connectivity products**: could affect single computers or even Ethernet cards but did not imply any problems in the total system;
- **network management** is still a problem, especially with bigger or complex networks.

A major MAP software manufacturer has recently announced an MMS-over-TCP/IP product. This could lead to a commercial breakthrough for MMS; running this application-layer protocol on top of the widely used TCP/IP protocol eliminates the need for complicated and expensive OSI products. If MMS-over-TCP/IP were to gain status as a *de facto* standard, integrating shop-floor control systems would be greatly simplified.

### 4 CONCLUSIONS

The three examples of CIM networking practices, presented above, show that multiprotocol Ethernet-based networks can be successfully used as an information highway for CIM on the shop-floor. Networking problems reported in the literature are recognized but do not imply serious problems for a successful implementation if systems are carefully designed.

### 5 REFERENCES


6 BIOGRAPHY

In 1989, Johan Putter received his Master’s Degree in electronics and electrical engineering from Chalmers University of Technology (Gothenburg, Sweden). From 1989 he is a Ph.D. student in the Department of Production Engineering at the same university. In 1993, he received the degree of Licentiate of Engineering. His research topics include computer integration in heterogeneous manufacturing environments and computer network technologies for open systems.
Events in CIMOSA and the CCE platform

S. Messina, P. Pleinevaux
Industrial Computing Laboratory, Department of Computer Science, Swiss Federal Institute of Technology, CH-1015 Lausanne, Switzerland
Tel: +41.21.693.67.97 Fax: +41.21.693.47.01, e-mail: silvia.messina@di.epfl.ch

Abstract
Event management is a mechanism useful for the specification of the system behaviour when specified conditions occur. The ability to represent reality in manufacturing systems may be enhanced by the availability of a well structured and powerful event management facility. We compare concepts like process, event, object, used to manage events, in the CIMOSA and CCE frameworks, two manufacturing modelling environments. We identify the relationship between such concepts. The two event management models are analysed in detail, they are compared and their differences are identified.

Keywords
Event management, manufacturing process modelling, integrating infrastructure, objects, information elements

1 INTRODUCTION

The aim of the present paper is to study the relationship between the CIMOSA and the CCE event models. We analyse the definition of their basic concepts, such as object, process and event in the CIMOSA framework and in the CCE platform. We compare their event management models and we identify the possible mappings between the two.

CIMOSA (Open System Architecture for CIM) (AMICE, 1994) provides a Reference Architecture for the modelling of a manufacturing enterprise. Its integration infrastructure offers a set of generic services for the execution of the enterprise model. The CCE (CIME Computing Environment) (CCE-CNMA, 1995) platform is an environment for the development, integration and execution of industrial applications. It has been developed as integration infrastructure of the CIMOSA platform. However, the two models have some differences and no exact mapping exists between their concepts. In the present paper we concentrate our attention on the event management models and their properties in CIMOSA and CCE.

Event management is a mechanism useful for the specification of the system behaviour under certain conditions. These conditions may be linked to physical devices, describing real events, or they may be triggered by user applications with the aim of synchronisation with or signalling to other applica-
The ability to represent reality in manufacturing systems may be enhanced by the availability of a well-structured and powerful event management facility. The need for such a facility in manufacturing environments is proved by the specification of event management mechanisms provided by existing industrial protocols (i.e., MMS (ISO/IEC, 1990)) and integrated manufacturing infrastructures (i.e., CIMOSA). CCE supports a limited event management.

The contribution of this paper consists in the comparison of concepts like process, event, object, used to manage events, in the two frameworks, CIMOSA and CCE, and the identification of the relationship between such concepts. The two event management models are analysed in detail, they are compared and their differences are identified.

Exceptions and exception handling should also be considered here for the sake of completeness, as extension of the present work, because of their commonalities with the event concept. Due to space limitation, we do not address here this issue and we recommend to the interested reader a preliminary study presented in (Messina, Pleinevaux, 1996).

The content of the paper is organised as follows: first an overview of the CIMOSA architecture and its event management is presented; the overview of the CCE platform and its event model follows, and a the relationship between CIMOSA and CCE is discussed. The second part of the paper presents the comparison and the mapping of the basic concepts of event management in CIMOSA and CCE: process, event, object, operation and attribute concepts are analysed in detail. The last section concludes the paper.

2 CIMOSA OVERVIEW

The CIMOSA Modelling Framework (AMICE, 1994) provides the necessary guidance to enable users to model the enterprise and its associated CIM system in a coherent way. The CIMOSA modelling approach is based on a Reference Architecture composed of reusable generic building blocks, which are aggregated to describe the enterprise model.

The CIMOSA model development is composed of three phases, starting with the Requirements Definition Modelling phase (AMICE, 1991). This model is described by the end-user that provides his view of the business needs. He gives his knowledge about the function, information and resources of the system. The next phases are the Design Specification and the Implementation Description. The example given below concentrates on the Requirements Definition Level, and we develop our proposal referring to the Requirements Definition Model of the enterprise.

The first step in the Requirement Definition Model development consists in the definition of the Domain to be modelled, its objective and constraints.

The Domain describes a part of the enterprise relevant for achieving defined set of business objectives. Examples of domains of activity in a real scenario are the Engineering Department or the Flexible Manufacturing System (Siemens, 1990), (Storr, et al., 1993).

Domains communicate among each other through events and describe the enterprise activities through Enterprise Objects and processes acting on them. An Enterprise Object is a generic entity of the enterprise that can be described by many Object Views. One Enterprise Object may be viewed from different points of view, thus it may correspond to several Object Views.

The functionality and the behaviour of a Domain is defined by Domain Processes. A Domain Process is a stand-alone process triggered by events and governing the execution of Enterprise Activities (the basic functionality) according to the so-called Procedural Rules. Each Domain Process is decomposed into Business Processes and/or Enterprise Activities. That is the Domain Process is decomposed into hierarchically structured functions, that are elementary functions. A set of Procedural Rules define the sequence of activation of Business Processes and/or Enterprise Activities.

An Enterprise Activity is detailed by describing its functionality, composed of several components,
some of which are:

- the Function Input (FI): set of Object Views to be processed and transformed by the activity;
- the Function Output (FO): set of Object Views produced or returned by the activity;
- the Resource Input (RI): is the possibly empty set of resources needed for the execution of the activity;
- the Resource Output (RO): textual statements indicating information to be recorded on the usage of the resources after the activity execution;
- the Control Input (CI): information used to control or constrain the activity execution;
- the Control Output (CO): set of Events generated by the activity;
- the Ending Status (ES): non-empty set of the possible termination statuses of the activity;
- the Activity Behaviour: finite algorithm specifying the functionality and behaviour of the activity; it is specified in terms of Functional Operations.

**Figure 1** Enterprise Activity definition.

The functionality of an Enterprise Activity is further decomposed at Design Specification Modelling level into a set of Functional Operations to be executed by Human, Machine or Application resources. The concept of Enterprise Activity corresponds to the concept of process, that may eventually be distributed on several hosts and remotely executed.

**CIMOSA event model**

Events in CIMOSA may be generated by Enterprise Activities, resources and external components in order to trigger Domain Processes. The CIMOSA event model is further analysed and compared with the CCE event model in Section 5.

### 3 CCE OVERVIEW

CCE (CIME Computing Environment) is an open environment for development, integration and execution of industrial applications. Its aim is to simplify the task of integration of applications in heterogeneous environments (CEC, 1993). This platform hides to the users the diversity in communication protocols, databases and access methods.

The CCE consists of an intermediate software layer between the operating system and the end-user application, a so-called middleware, available on various hardware and software environments, providing a complete platform for the development, integration and operation of manufacturing applications. CCE is aimed at making the applications independent from the hardware and software environment in which they run: this environment is composed of computers, operating systems, networks, industrial devices, databases, proprietary and standard applications, etc.

The CCE architecture follows a client/server model: a client requests a CCE service through a CCE application programming interface (API), and a dedicated CCE server executes the service and sends
the response back to the requesting application. The roles of client and server of the CCE components
may change during the provision of the service: whereas CCE applications or the CCE administration
are client applications when calling the CCE services, a CCE server itself may need to call another
CCE server in order to provide the required service. A CCE server may also behave as a client with
respect to servers outside the execution environment, such as MMS servers on automation systems,
database servers or file servers.

The CCE object model
An object-oriented approach has been chosen to hide the differences and the complexities of the
various data accesses and services to the application developer. An object represents something
that has a counterpart in the real world (a device, a program, a tool, a pallet, etc.). It is specified
by three sets of features (see Figure 2): attributes which characterise the object, operations

Figure 2 The CCE object description.

which can be applied on the object and event notifications which are sent by the object.
For example (see Figure 3), a 'program' object can have the attributes 'name', 'state', 'list of domains',

Figure 3 The PROGRAM object description.

the operations 'download', 'execute', and the event notifications 'downloaded' and 'end of execution'.
The object interface allows to access and modify the object attributes, to invoke the operations and to
subscribe to the event notifications sent by the object. An object is implemented by a server which pro-
vides all the services defined at its interface and detects and notifies events specific to this object.
The CCE event model
In the CCE model, events are transmitted as *notifications* and are associated with the CCE objects. They are sent to the client applications by CCE objects. Notifications are part of the object description, they are not modelled as a distinct object. The client application must subscribe to the notifications it wants to receive.

A possibly empty set of pre-defined notifications is specified for each object type. Each notification message has a fixed format. No new notifications can be defined on existing CCE objects (Silicomp, 1996).

The triggering of CCE event notifications is only internally monitored. Client applications may subscribe to the notifications they are interested in. Functional servers, access servers, information servers and processes may subscribe to notifications as well. The object that sends the notification takes care of sending it to all the subscribing CCE components (applications and servers), following a producer/consumer model. Thus, the producer must know the identity of all the consumers.

4 RELATIONSHIP BETWEEN CCE AND CIMOSA

The CIMOSA model (AMICE, 1994) is produced through a process composed of several phases, referred as "stepwise generation": the model is generated by identifying successively the requirements, design and implementation needs, in any appropriate order and iterating as necessary to achieve optimal solutions.

Figure 4 shows the system development cycle defined according to the software engineering terminology (Pfleeger, 1987). The CIMOSA model generation process covers all the four phases, while the CCE platform is used only at implementation phase. CCE and CIMOSA concentrate on the design of the software architecture for manufacturing applications. CCE was inspired by CIMOSA, and intends to provide implementations of the physical and application integration framework.

CIMOSA identifies four enterprise views to model the major aspects of an enterprise independently of each other, namely the function view, the information view, the resource view and the organisation...
view. These views are not all present in the CCE model and are not so clearly separated. The basic component of the CCE model is the object, that can represent an information or resource entity, and at the same time it can partially express functional information, represented by the operations defined on the object. The CCE operations do not represent entirely the function view, since the function view includes not only the functionality description but also the control information for the execution of the operations. Thus the function, information and resource views appear as intrinsically, even if partially, supported by the CCE object model. To our knowledge the organisation view has not yet been considered by CCE (see Table 1).

Table 1 CIMOSA views and CCE components relationship

<table>
<thead>
<tr>
<th>CIMOSA views</th>
<th>CCE components</th>
</tr>
</thead>
<tbody>
<tr>
<td>function</td>
<td>partially supported by CCE operations + CCE notification</td>
</tr>
<tr>
<td>information</td>
<td>CCE object</td>
</tr>
<tr>
<td>resource</td>
<td>represented by CCE object</td>
</tr>
<tr>
<td>organisation</td>
<td>not represented</td>
</tr>
</tbody>
</table>

The CIMOSA integrating infrastructure hides the location, storage and physical placement of information and resources from the requestor of the information and manages the link with the underlying communication infrastructure. The CCE platform has similar objectives and can be used as CIMOSA integrating infrastructure (Pleinevaux, 1996).

CCE offers services that comply with the definition of CIMOSA integrating infrastructure services, namely common, presentation, information and business services (Pleinevaux, 1994). CCE covers the requirements of common, presentation and information services. Only the business services are not covered by the CCE platform. These services are defined in CIMOSA for the execution of the models of the enterprise. In CCE, application knowledge is mainly provided to the system in the form of programs, not in executable models.

5 CCE AND CIMOSA: A COMPARISON OF THE BASIC CONCEPTS

In this section we analyse the definition and use of events and of concepts such as processes and objects, as they are defined in CCE and CIMOSA. We discuss their differences and similarities.

5.1 Events and processes

The events defined in CIMOSA can be compared with the CCE event notifications. Their apparent differences are mainly due to a simplification of the event model realised at implementation phase.

CCE events are sent by objects to the applications that have subscribed to them. CIMOSA events are sent by Enterprise Activities, resources or external components to Domain Processes. Thus, apparently, a difference exists between the event producers and consumers in CCE and CIMOSA.

Let us consider first the event consumers. In the CCE model the applications are the only active entities that can send operation requests and receive notifications from the objects.

The CIMOSA concepts of Enterprise Activity, Business and Domain Process are implemented in CCE in the application programs. We call ‘CCE process’ the execution of a CCE application program. We consider a process an active entity, able to communicate with other processes, and asynchronously exe-
A CCE process is composed of sub-processes (and sub-sub-processes, at several levels of decomposition) or subprograms and simple operations. This hierarchy may be mapped onto the CIMOSA Enterprise Activity, Business and Domain Process hierarchy in several ways: no unique mapping exists and the implementor decides how to structure his application programs. The most natural mapping is the one that establishes a match between the Domain Process and the CCE process. As consequence of this choice, a mapping exists between the Business Processes and the sub-processes or subprograms composing the CCE process; and a mapping exists between the Enterprise Activities and the sub-sub-processes or the simple operations composing the CCE sub-processes. With this approach the Domain is mapped to a set of CCE processes. Table 2 summarises this example of mapping between CCE and CIMOSA.

**Table 2** One possible mapping between CIMOSA and CCE process concepts

<table>
<thead>
<tr>
<th>CIMOSA concept</th>
<th>CCE concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>set of CCE processes</td>
</tr>
<tr>
<td>Domain Process</td>
<td>CCE process (application execution)</td>
</tr>
<tr>
<td>Business Process</td>
<td>application sub-process or subprogram</td>
</tr>
<tr>
<td>Enterprise Activity</td>
<td>sub-sub-process or subprogram or operation</td>
</tr>
</tbody>
</table>

However, as we said, the implementor may choose to map in a different way its application programs: he can represent the entire Domain as a single CCE process, the Domain Processes as the sub-processes of this CCE process, the Business Processes as the sub-sub-processes and the Enterprise Activities as the procedures or operations. To the opposite extreme, an example of mapping is shown in Table 3, as third option.

**Table 3** Mappings between CIMOSA and CCE process concepts

<table>
<thead>
<tr>
<th>CIMOSA concept</th>
<th>Mapping 1</th>
<th>Mapping 2</th>
<th>Mapping 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>CCE process</td>
<td>set of CCE processes</td>
<td>set of sets of CCE processes</td>
</tr>
<tr>
<td>Domain Process</td>
<td>sub-process</td>
<td>CCE process</td>
<td>set of CCE processes</td>
</tr>
<tr>
<td>Business Process</td>
<td>sub-sub-process</td>
<td>sub-process</td>
<td>CCE process</td>
</tr>
<tr>
<td>Enterprise Activity</td>
<td>operation</td>
<td>sub-sub-process/operation</td>
<td>process/sub-process/operation</td>
</tr>
</tbody>
</table>

CIMOSA designers propose one possible mapping of IDL (Implementation Description Language) constructs onto corresponding CIMOSA constructs, where Business Processes are mapped onto parallel processes, Enterprise Activities onto sequential statements and Functional Operations onto expressions (AMICE, 1994).

A fundamental difference between the CCE and the CIMOSA models appears in this context. The CCE application programs are composed of subprograms and operations, that are defined by the user as ASPI calls. The execution of a CCE application is controlled step-by-step by the user by calling the CCE interface, and getting the results of the execution from the platform. The CIMOSA model specification made by the user allows to define the Enterprise Activities, the Processes and the Domains, at a higher level of abstraction. The model should be mapped to executable processes. If this mapping is realised (this is a target not completely satisfied by the CIMOSA project), the CIMOSA infrastructure takes care
of the execution of the processes and their sequence, at all levels from Functional Operations up to the Domain Processes, without the user being responsible for controlling their execution.

In conclusion, we see that the consumer of CIMOSA events, the Domain Process, corresponds in mapping 2 to the CCE event consumer, the application program (CCE process). The mapping is shown in gray in Table 3.

Now let us consider the event producers. CIMOSA events can be generated by enterprise resources, Enterprise Activities or external components. These three types of event producers may trigger event conditions. In CCE, events are sent by objects when pre-defined and internally monitored conditions occur. These conditions may be modified by user application requests or by physical events. User applications require operations on the objects and these operations may modify the monitored object conditions and generate events. Thus one source of events in CCE is the operation that causes the triggering of the event condition. Objects cannot directly trigger their own event condition, nor the event conditions of other objects. CCE objects may model physical devices or physical entities, e.g., a CCE object variable may model the physical value measured by a sensor. The physical device or entity state changes are reflected into the object state and may modify the monitored object conditions. Thus sources of events in CCE may also be the physical events. External applications, not modelled by CCE, may be sources of events as well, and may send events to CCE applications. Thus, CCE event producers are operations on objects, physical devices or entities and external applications. Table 4 summarises the mapping. CCE Event conditions are only internally monitored, and are detected by polling.

Table 4 Mapping between CIMOSA event producer and CCE event producer

<table>
<thead>
<tr>
<th>CIMOSA event producer</th>
<th>CCE event producer</th>
</tr>
</thead>
<tbody>
<tr>
<td>resource</td>
<td>physical device</td>
</tr>
<tr>
<td>Enterprise Activity</td>
<td>operations executed by user applications</td>
</tr>
<tr>
<td>external component</td>
<td>external application</td>
</tr>
</tbody>
</table>

Another point that may seem to represent a difference is the triggered action. CIMOSA does not say explicitly anything about it, but the Domain Process, consumer of the event, acts as triggered action, because it is executed as consequence of the event occurrence. Also CCE allows to associate triggered actions with event occurrences, but these actions are limited to single operations on the same object where the event occurred and the actions are pre-defined in the platform definition. In the current version of CCE, the user cannot define his own actions. Thus we see that the CCE event model is also in this case more limited than the general specification provided by CIMOSA. Table 5 summarises the mapping of event concepts in CIMOSA and CCE.

Table 5 Mapping between CIMOSA and CCE event models

<table>
<thead>
<tr>
<th>event concept</th>
<th>CIMOSA RS Model</th>
<th>CCE Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>producer</td>
<td>enterprise resource +</td>
<td>physical entity +</td>
</tr>
<tr>
<td>Enterprise Activity</td>
<td>+</td>
<td>operation +</td>
</tr>
<tr>
<td>external component</td>
<td></td>
<td>external application</td>
</tr>
<tr>
<td>consumer</td>
<td>Domain Process</td>
<td>process (application execution)</td>
</tr>
<tr>
<td>event action</td>
<td>Domain Process</td>
<td>operation on object</td>
</tr>
</tbody>
</table>
We write in bold style the concepts where a difference exists between the two models. The difference always consists in a more limited functionality supported by CCE.

Other important event issues that must be discussed are: event conditions, event subscriptions, polling of conditions, priority, severity, suspend/resume monitoring. These issues are implementation dependent and they are not specified explicitly by CIMOSA. Another interesting feature is the acknowledgment of notifications, that allows checking the receipt by the subscribers. It is not specified by the CIMOSA model and it is not supported by CCE.

Event conditions have priorities that are used to schedule their processing according to the importance of the event relative to other events; they have severity to represent the effect of the event on the process which is being controlled. The event condition polling may be temporarily suspended and then resumed. CIMOSA provides event condition specification in the form of natural language sentences; subscription to events is implicit in the model, no explicit subscription action is defined by CIMOSA. The other issues are not taken into consideration by the CIMOSA Requirement Specification Model, since they are mainly design or implementation issues.

The CCE platform, on the other hand, provides a possible implementation of the flexible concepts of CIMOSA, thus it does implementation choices that limit the flexibility and generality of CIMOSA: no event condition definition is allowed by CCE, an explicit operation for event subscription is defined. Applications may specify the event polling period. Event notifications may have a specified priority, but no severity. No suspend/resume monitoring option is provided and no event pulling by the client application is allowed.

In conclusion, we think that the CCE platform implements a limited CIMOSA concept of event, imposing restrictions sometimes due to implementation constraints, other times due to a simpler event model.

### 5.2 Objects and information elements

The CIMOSA objects are generalised concrete or abstract entities of the enterprise. CIMOSA objects model enterprise resources (human, machine or application), and resources are one of the possible sources of events. Objects are defined by the users during the Information Analysis to identify the objects involved in all the Enterprise Activities. Objects may be viewed from different points of view by the users or applications. Thus each object may have several Object Views. Objects are described by either Information Elements or lower-level objects (also called sub-objects). Information Elements are the attributes of the objects. Integrity rules may be defined on Information Elements, in order to impose constraints used to ensure the validity and correctness of the Information Elements (e.g., domain constraints, consistency constraints, and so on). CIMOSA objects are defined as entities used for the execution of the Enterprise Activities and Business Processes, manipulated by them, shared by several Enterprise Activities and Processes.

CCE objects, like CIMOSA objects, model enterprise resources and can send notifications. But CCE object classes are pre-defined by the CCE platform: the user can only create objects belonging to the pre-defined classes. Each object has a unique view; no concept of several object views exists for CCE objects. The definition of CCE objects is provided by the CCE platform, in terms of attributes, operations and notifications. In the current version of CCE, the user cannot modify the object definition, nor define a new class. He cannot modify the event notification or define a new one. The reason is that all these operations require a good understanding of the platform internal.

In conclusion, CCE represents one possible implementation of the CIMOSA objects, but more limited, since it does not support object views, integrity constraints on object attributes and it does not allow the dynamic definition of events. Table 6 summarises the results of the comparison.
Table 6 Mapping of CIMOSA and CCE object concepts

<table>
<thead>
<tr>
<th>CIMOSA</th>
<th>CCE</th>
<th>comparison result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise Object + Object Views</td>
<td>object</td>
<td>CCE supports only one Object View for each object</td>
</tr>
<tr>
<td>Information Element + Integrity Rule</td>
<td>attribute</td>
<td>no integrity constraints on attributes are supported by CCE</td>
</tr>
<tr>
<td>Enterprise Activity and Business Process share objects</td>
<td>operation</td>
<td>CCE operations act on the object on which they are defined</td>
</tr>
<tr>
<td>dynamic definition of events generated by resources</td>
<td>pre-defined notifications sent by objects</td>
<td>CCE does not support dynamic definition of events</td>
</tr>
</tbody>
</table>

6 EXAMPLE

Let us consider the following example. Within a real manufacturing scenario, we take into consideration a Flexible Manufacturing System Domain. Inside this domain, we consider the Part Production Domain Process, which deals with the manufacturing process. This Domain Process is composed of the following Business Processes and Enterprise Activities:

Domain Process: Part Production
  Business Process: Input Raw Material
  Business Process: Toolset Preparing
  Business Process: Machining
    Enterprise Activity: Support Preparing
    Enterprise Activity: Manufacture Part
  Business Process: Machining Inspection
  Business Process: Output Parts

The Business Process ‘Machining’ represents the manufacturing operations that are executed on the raw material in order to obtain the manufactured parts. The component Enterprise Activities decompose the machining process into more elementary steps. The Enterprise Activity functionality description is as follows:

Enterprise Activity: Manufacture Part
  FI: Tool Data, Raw Material
  CI: Part Program
  RI: Machine Tool, Tools, Raw Material
  FO: Part
  CO: Event Notification: End of Manufacturing
  RO: Tool Data
  ES: DONE
  Activity Behaviour: Download Part Program and Tool Data, Start Part Program, Send ‘End of Manufacturing’ event notification Upload Tool Data
The Business Process functionality description is as follows:

**Business Process: Machining**

**Procedural Rules:**
- WHEN Start DO BP Input Raw Material
- WHEN ES(Input Raw Material)=DONE DO BP Toolset Preparing
- WHEN ES(BP Toolset Preparing)=DONE DO BP Machining
- ........
- WHEN ES(BP Output Parts)=DONE DO FINISH

The 'Manufacture Part' Enterprise Activity functionality is described by the sequence of Functional Operations. These are mapped at implementation phase to a CCE process. This CCE process is composed of a sequence of CCE operations on objects, thus a mapping is defined between each Functional Operation of the Enterprise Activity and each CCE operation, as shown in Table 7. The shortness of this example is imposed by space limitations.

<table>
<thead>
<tr>
<th>Functional Operations</th>
<th>CCE operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Download Part Program and Tool Data</td>
<td>CCE_Download (Part Program; Tool Data)</td>
</tr>
<tr>
<td>Start Part Program</td>
<td>CCE_StartProgram</td>
</tr>
<tr>
<td>Send 'End of Manufacturing' event notification</td>
<td>CCE_Receive(Event Notification)</td>
</tr>
<tr>
<td>Upload Tool Data</td>
<td>CCE_Upload(Tool Data)</td>
</tr>
</tbody>
</table>

7 CONCLUSION

The goal of this paper was the analysis of the relationship between the event management models in the CCE and the CIMOSA architectures. We have first introduced the CIMOSA and CCE models, discussed their relationship and then analysed the mapping existing between the concepts defined in their respective event models.

From the above discussion, we can conclude the following:
- CCE objects are active (they may send notifications), as are CIMOSA resource objects; some differences in the two event models are mainly due to a simplification of the CIMOSA event model realised at implementation phase: the CCE platform implements a limited CIMOSA concept of event, imposing restrictions sometimes due to implementation constraints, other times due to a simpler event model. Main limitations are:
  - the mapping between CIMOSA event consumers and CCE event consumers is left to the implementor choice;
  - CCE limits the event generation to the indirect modifications acted by operations (or procedures) on object states;
  - CCE does not support event triggering;
  - CCE event actions are pre-defined; no event condition definition is supported by CCE;
  - CCE does not support object views and integrity constraints on attributes;
  - no dynamic creation/modification of CCE object classes is allowed.
- The only fundamental difference between the CCE and the CIMOSA models is the level of ab-
straction: the CCE application programs are considered as sequences of service calls to the platform interface, each call being a single CCE operation, while the Domain Processes, Business Processes and Enterprise Activities may model complex behaviours that should theoretically be transformed by the platform into executable processes.

The analysis presented in this paper represents the first step of a future study devoted to the identification of the commonalities and differences between the CIMOSA and CCE architectures. The future work aims at the definition of a mapping tool of all CIMOSA and CCE concepts.

8 REFERENCES

Deliverable to CEC (1993) CCE Implementation Guide - Revision 1.0

9 BIOGRAPHY

S. Messina graduated in Computer Science at the University of Torino (Italy) in 1989. She received her Ph.D. in Computer Science from the Swiss Federal Institute of Technology in 1996. She has been an assistant/researcher there, in the Industrial Computing Laboratory, since 1990. Her interests include enterprise modelling, object-oriented approaches and business engineering.

P. Pleinevaux obtained a degree in Electrical Engineering in 1984 from the University of Brussels and a Ph.D. in Computer Science from the Swiss Federal Institute of Technology in 1990. He was technical manager of the ESPRIT CNMA and CCE-CNMA projects between 1991 and 1995. He is a lecturer in industrial computer engineering at the Swiss Federal Institute of Technology in Lausanne.
Adaptable Low Cost Shop-Floor Control System for Central and East European Companies

J. Uhl
ISW, University of Stuttgart
Seidenstr. 36, 70174 Stuttgart, Germany
tel. +49 711 121 2343, fax: +49 711 121 2413
e-mail: joachim.uhl@isw.uni-stuttgart.de

Z. Smalec, J. Chrobot, J. Rakowski
ITMiA, Technical University of Wroclaw
Lukasiewicza 3/5, 50-371 Wroclaw, Poland,
tel. +48 (71) 320 38 78, fax: +48 (71) 21 06 70,
e-mail: {zbyszek, robot, rakos}@itma.pwr.wroc.pl

Abstract
Industrial production in Central and East European countries is characterised by low productivity. An improvement of the situation would be achieved by a more effective application and use of existing resources and equipment as well as introducing a computer-aided information flow. Shop-floor control systems play an important part as solution to these problems. This paper contains the goals and current results of the Copernicus project „Adaptable Low Cost Shop-Floor Control System“ supported by the European Community. The goal of this project is to develop a low-cost shop-floor control system which should be adaptable in many different Central and East European companies. The first stage of this project has already been finished. The paper contains conclusions after implementation of this project stage.

Keywords
shop-floor control system, control system, software engineering, adaptation methods, low-cost
1 INTRODUCTION

Shop-floor control systems can be assigned to the Computer Aided Manufacturing Area according to Figure 1 (Pritschow, Uhl, 1995). Shop-floor control systems' tasks are the capacity planning of production units, the detailed planning of orders released by the Production Planning and Control (PPC), release of orders, as well as recording and evaluation of responses from the shop. With this the rough scheduling stated by the PPC is carried out and compressed data about the orders process are returned to the PPC (Eversheim, 1990; Nedeß, 1993).

Since changing to a free-market company, there is a need for effective production coordination and monitoring by shop-floor control systems in Central and East European companies. Unfortunately, the market offers above all expensive shop-floor control systems having large extent of functions.

The contents of this article are therefore concerned with the structure and development of a shop-floor control system in an EC supported project that corresponds to the requirements of Central and East European companies.

![Figure 1: Categorisation of shop-floor control systems.](image-url)
2 PROBLEM STATEMENT AND REQUIREMENTS OF A SHOP-FLOOR CONTROL SYSTEM FOR CENTRAL AND EAST EUROPEAN COMPANIES

2.1 Problem statement in Central and East European companies

Decades of planned economies and lacking market-economy thought in Central and East European companies led to the following problems which have to be considered in the introduction and use of shop-floor control systems:

- The current production companies came, for the most part, from large centrally controlled combines. The out-dated and hard-to-understand organisation structures and information flow in the companies have remained intact due to rigid structures and are now slowly changing. A reorganisation for improving company processes and information flow is usually necessary before introducing control systems.

- A wide use of computer applications, as is common in all areas in Western European companies, is not present in East European companies. This is also true to the same extent for the level of worker training in using computer applications. An introduction of shop-floor control systems must therefore generally be in connection with intensive training of the workers.

- A chronic lack of capital does not allow for the purchase of shop-floor control software that fits the requirements, and therefore expensive, from Western European companies.

2.2 Problems with the introduction and use of purchasable shop-floor control systems in Central and East European companies

Investment in capital-intensive plants to be able to have high quality production, requires effective planning, control and monitoring of production. The last part requires use of software solutions for shop-floor control systems. However, the introduction and use of purchasable shop-floor control systems has the following problems:

- Good-value, requirement-sized PC solutions with low functionality and that are highly modifiable are not available in the West.

- Shop-floor control systems are usually not offered in the relevant local language or are poorly translated. They are usually not user-friendly, so the user needs a considerable amount of time to become familiar with them as hardly any basic knowledge can be expected.

- The shop-floor control system introduction costs from Western European companies are very high. East European marketing companies sometimes have neither the respective systems nor sufficient knowledge for an integration of the shop-floor control system.

- In Central and East Europe the information flow and the organisation of companies from the same product area differ more than in the West, therefore more adaptation of a shop-floor control systems is necessary.
• An integration of shop-floor control systems usually proves to be difficult, especially when software solutions which already exist in the companies, must be considered. This is because either self-developed solutions or simply closed software solutions were implemented which do not have the required interfaces.

• Organisational improvements and product technology improvements require a high flexibility, adaptability and extendibility of a shop-floor control system, which is generally not currently available.

2.3 Requirements of a shop-floor control system for Central and East European companies

For the above problems, requirements of a shop-floor control system for Central and East European companies can be derived from Figure 2.

![Figure 2: Requirements of a shop-floor control system for Central and East European companies.](image)

An important criteria for the shop-floor control system is low costs. This means low costs development, the costs which influence the licensing costs for the shop-floor control system, costs of integration and introduction as well as the necessary hardware and software tools. Therefore a software solution is planned with limited functionality with only the minimum necessary functionality. Using the shop-floor control system on a PC is strived for with limited costs for the additionally needed software, for example, for the operating system or database.

User interfaces and callable functions must be kept simple and self explanatory. A simple software adaptation of the shop-floor control system must be guaranteed. Beside the simple adaptability on various operating systems and hardware environments, this also concerns an
adaptability to various organisational structures and information flows. A change by improving the structures must be considered especially. For this, various adaptability methods (Brantner, 1993; Siewert, 1994; Pritschow and Uhl, 1994) should be used.

Additionally, a systematic procedure must be defined for an initial development as well as adaptation development for integrating and set-up of software for a shop-floor control system. This is needed to be able to carry out a quick, simple and inexpensive set-up of the shop-floor control system based on company specific requirements and the results of a company analysis, for example by parameterising and especially also adaptability development for integration (Pritschow and Uhl, 1994; Siewert, 1994). This has to be done in a repeatable manner so that the master control system is not the individual expertise of the respective developer, but rather that adaptation developments may also be carried out by reasonably priced workers.

3 FUNCTIONS OF THE SHOP-FLOOR CONTROL SYSTEM

According to Figure 3, the functions of a shop-floor control system can be differentiated as the following functions:

- managing
- planning
- operative.

![Figure 3: Functions of a shop-floor control system.](image)

The functions are each shown separately in function modules. The structuring occurs in a way that the modules are, to a large degree, autonomous and can operate in parallel. With this
Adaptable shop-floor control system

modular formation it is possible to independently use single components or to swap out components for changed requirements.

Characteristic functions of a shop-floor control system are:

- order management: management of production orders with piece numbers, planning horizon and tracking order advancement;
- process plan management: management of work-piece related process plans and the order of production steps, production alternatives and affiliated manufacturing accessories;
- capacity planning: detailed planning of production orders on machines or manual work places, creation of (machine) load plans;
- personnel calendar: management of shift plans;
- manufacturing accessories scheduling: order specific demand of manufacturing accessories such as e.g. tools, pallets and fixtures, manufacturing accessories set-up and availability check;
- manufacturing accessories management: management of master data for types of manufacturing accessories types and data for single manufacturing accessories;
- worker-team assignment: order release and order assignment to teams and individual workers, simple order processing;
- production data acquisition: recording, protocolling, and forwarding of process reports, display of plant status, interruptions and transporting, evaluation of messages;
- NC data activation: loading and receiving NC programmes, corrections and zero point changes to the NC machines, starting the program;
- NC data management: management of NC programs and offset values of tools and pallets.

4 ADAPTABILITY METHODS OF A SHOP-FLOOR CONTROL SYSTEM

Adaptability methods are aids and mechanisms which make it possible to fit a shop-floor control system to the requirements of another company without changing the software or by only making small changes, or developments, to the software. By using adaptation methods the introduction of a shop-floor control system limits itself extensively to one initiation (see also chapter 5). This way, further developmental work on the shop-floor control system is avoided. Adaptability methods have to be considered in the development of a shop-floor control system. Figure 4 shows a list of adaptability methods.

Standards simplify the portability of software, especially user interfaces and database access on other computers or on other operating systems.

Development platforms with software libraries free the software engineer from routine programming and he may use tested software which is of higher quality. Additionally, operating system specific functions and database specific functions are encapsulated. Also, using a development platform and software libraries, portability is increased. This especially
concerns procedure calls for communication between function modules as well as other operating system calls in function modules.

**Configuration** simplifies the step by step extension of a shop-floor control system with function modules and allows a simple exchange or an individual assembly of the master control system according to needs.

**Parameterising** data increases the flexibility and the area of use of the function modules without needing changes in the software of the function module.

**Adaptation modules** simplify the connection of the shop-floor control system with other systems, i.e. a PPC-system, without changing the software of the shop-floor control system. Software development work is needed for adaptation modules for new uses.

![Diagram](image)

**Figure 4:** Adaptability mechanisms for shop-floor control systems.

## 5 PROCEDURES FOR AN INITIAL DEVELOPMENT OR ADAPTATION DEVELOPMENT AS WELL AS SET-UP

The following chapter describes the procedure for developing and setting up the shop-floor control system. It is shown how a **systematic continuous procedure** saves **time and costs** in the initial development and in the following use of the shop-floor control system. It also shows how a reproducible, documented development process the integration and set-up of the shop-floor control system is simplified and no longer must be carried out by the developer of the shop-floor control system.
**Requirements analysis**
In the requirements analysis (see Figure 5) the organisational structure and the information flow of the individual company are investigated. This occurs with emphasis on the area or areas in which the shop-floor control system is to be introduced. The goal of the requirements analysis is to make:

- company schemes and
- process chain diagrams

of the company and the affected areas. This model was represented by a self developed notation (Copernicus, 1995). The notation contains:

- company schemes for hierarchical representation of the organisation structures and information flows;
- process chain diagrams in order to show functions, data and processes within the organisation units defined in the company schemes.

![Phase of the requirement analysis](image)

**Figure 5:** Requirement analysis phase.

From this model one can:

- derive the required data flow and functions as well as interfaces by comparing it to a reference model for the developed shop-floor control system and in agreement with the client;
- determine the weak points in the organisation and information flow in order to incorporate suggestions for improvement.
System analysis
Starting from the company schemes and the process chain diagrams from the requirement
analysis, in the system analysis an abstract model of the software is constructed in agreement
with the client. As a description method (see Figure 6) the following is selected:
- structured analysis describing the functions of the data flow and the interface of the shop-
  floor control system (DeMarco, 1978);

Data flow and the essential functions may be derived from the company schemes and process
chain diagrams. In resulting uses the software analysis model serves agreements with the client
and to determine the possible expansion development or adaptation development.
Configuration and parameter possibilities as well as the use of adaptability functions are
established in this phase.

![Phase of the system analysis](image)

**Figure 6:** System Analysis Phase.

System design, coding and testing
The goal of this phase (Figure 7) is to convert the functions of the shop-floor control system
and data flows from the system analysis into computer processes and to design these
processes. For description methods the following are used:
- process diagram for describing the computer processes of the shop-floor control system;
- structured design for describing the computer processes of the process diagram
  (Constantine and Yourdon, 1979);
- the Nassi-Shneiderman-structograms for describing procedures from the Structured design
diagram (Nassi and Shneiderman, 1973).
The process diagram here is a self developed description method which has also already been agreed upon on the development platform (see chapter 4) of the developed shop-floor control system (Copernicus, 1996). The computer processes and the telegram exchange between the computer processes were taken from the essential functions as well as from the data flow between these functions of the software model from the software analysis phase. In this phase those procedures are also defined which encapsulate the database access and the operating system access and assign them to a replaceable software library.

![Diagram of the system design phase](image)

**Figure 7:** System design phase.

**Setting up phase**
When setting up, in addition to installation of the shop-floor control system, the software adaptation takes place. In this phase the shop-floor control system is configured and the individual modules used are parameterised. For adaptation to the computer, relevant software libraries must be integrated for operating system and database.

### 6 IMPLEMENTATION EXAMPLE

The three-year long, Copernicus project (no. CP9400337), started in March 1995, deals with the shop-floor control system described in the previous chapters. Three institutes are participating in the project. These are: the Institute for Control Technology for Machine Tools and Manufacturing Systems (ISW) of the University of Stuttgart in Germany, the Institute of Mechanical Engineering and Automation (ITMiA) of the University of Wroclaw in Poland, and the Institute for Machine Tools of the University of Prague in the Czech Republic. The companies PZL Hydral of Poland (producer of hydraulic pumps for general and specific uses), ZDAS of the Czech Republic (producer of rolling mill equipment and metal-forming...
machines) and PPS Detva of Slovakia (producer of fork-lift trucks among other things) are taking part in the project as industry partners and future users.

For example, at PZL Hydral the shop-floor control system shall be implemented in the production division for case machining (see Figure 8). Here the analysis of the organisation and information flow has already led to improvements such as the introduction of a validity period for technical documentation. It was also made clear that the use of the DNC is not yet sensible because of completely absent, or only partly available, DNC capable controllers.

Figure 8: Case Machining Department at PZL Hydral, Poland.

Figure 9: Graphical user interface for capacity planning and scheduling.
Figure 9 shows a draft example of user interface of the workshop-control system. The figure illustrates the graphical user interface for capacity planning, which allows scheduling of orders for planning and release of processes to workers and teams assignment functions. The scheduling could be carried out manually, using graphical methods. There are also functions for capacity planning and process sequencing and also functions that allow work station utilisation and assembly line balancing to be reviewed.

7 SUMMARY

The preceding report shows the empirical values for developing and introducing a shop-floor control system for Central and East European companies within the EC-supported Copernicus Project. It became clear that in Central and East European companies only low priced, simply constructed and integratable and adaptable shop-floor control systems can be used. This is obtained by a PC based shop-floor control system structured in accordance with function modules and having a minimally needed functionality, and by using adaptability methods as well as with a systematic, reproducible development process. With this project it is possible to cover the needs Central and East European companies in accordance with a simple software support for the control and monitoring of their production, where simultaneously a transfer of expertise to East European engineers takes place.

8 REFERENCES


9 BIOGRAPHY

Joachim Uhl holds a master's degree in mechanical engineering. Since 1990, he has been working as research assistant at the Institute of control technology (ISW) at the University of Stuttgart. He is head of the group 'production control systems and quality assurance' and head of the COPERNICUS project 'Adaptable low-cost shop-floor control system' funded by the EC. His main research interests are the application of object-oriented techniques in flexible manufacturing, the development of decentralised control systems and the development of open system architectures for cell controls.

Zbigniew Smalec has been holding a doctor's degree in mechanical engineering since 1982. Since 1991 he has been senior lecturer at the Department of Machine Tools and Automation at the Institute of Mechanical Engineering and Automation (ITMiA), Technical University of Wroclaw, Poland. Since 1994 he has been member of Polish Association of Mechanical Engineers. He is head of Research Group for Control and Technology at the Institute of Mechanical Engineering and Automation. His scientific activity are overall about 40 publications in journals and conferences and numerous research projects for industry. His main research interests are tool machining processes, the application of industrial communication nets and control systems.

Jaroslaw Chrobot holds master's degree in automation and robotics. In the year 1992-1993 he has studied at the University of Stuttgart, Germany. Since 1994 he has been PhD student at the Institute of Mechanical Engineering and Automation, Technical University of Wroclaw (Poland) in the Research Group for Control and Technology. He has been working on the COPERNICUS project 'Adaptable low-cost shop-floor control system' funded by the EC. His main research interest is development of shop-floor control systems.

Jaroslaw Rakowski holds master's degree in automation and robotics. In the year 1992-1993 he has studied at the University of Stuttgart, Germany. Since 1994 he has been PhD student at the Institute of Mechanical Engineering and Automation, Technical University of Wroclaw (Poland) in the Research Group for Control and Technology. He has been working on the COPERNICUS project 'Adaptable low-cost shop-floor control system' funded by the EC. His main research interest is application of shop-floor control systems.
Global Networks and Their Applications
Information infrastructure services for small and medium size manufacturers: The MI²CI project

J. Goossenaerts\textsuperscript{a}, C.M. Acebedo\textsuperscript{b}
\textsuperscript{a} Eindhoven University of Technology, Information & Technology
P.O.Box 513, Paviljoen, 5600 MB Eindhoven, The Netherlands
\textsuperscript{b} De La Salle University, Industrial Engineering Department
2401 Taft Avenue, 1004 Manila, Philippines

Abstract
Advanced information and communication technologies have become major enablers of manufacturing industry operations and product and process development.

The paper proposes the concept of an information and command infrastructure and the role it can play as an enabler for lean, agile and sustainable industries in developing countries. An information infrastructure should provide an ongoing and lasting stream of information, decision and control services in support of the different life cycle phases of products and production resources, i.e. their design, production, use, distribution and disposal. Such services are particularly important for small and medium size manufacturers.

The MI²CI project deals with software technology for information and command infrastructures for industries. Its goals and current activities are described.

Keywords
Industrial development, information infrastructure, enterprise integration, small and medium size enterprises

1 INTRODUCTION
During the last decades the value added to raw materials through manufacturing has been increasing far more than the value of the raw materials, energy and agricultural goods
traded on the world market. Because of this global market tendency developing countries feel a growing pressure to catch up in manufacturing technology. However, the growing variety of manufactured goods and the increasing expertise and know-how required for producing them, hamper these countries in achieving their industrial development goals, also in areas where small and midsized manufacturers play a substantial role. Meanwhile, the opinions are gaining acceptance that markets should be open and industrial production environmentally sustainable. By including these additional requirements in their industrial development agenda, many developing countries face an even larger catch-up hurdle: rather than starting from "simple" low-tech mass production, develop human resources, eliminate waste, improve skills, and increase product variety, developing countries are now expected to enter their industrial age at the levels of global competitiveness and environmental sustainability.

Having in view the development difficulties of emerging industries, meanwhile observing that manufacturing industry operations and product and process development are increasingly being executed by lean and agile enterprises and by extended and virtual enterprises, supported by advanced information and communication technology, UNU/IIST\(^1\) has initiated the \textit{MI}^2\textit{Cl} (Manufacturing Industry Information and Command Infrastructure) project. This project, which has been endorsed by UNIDO\(^2\), information and communications technologies based software applications that can support emerging manufacturing industries in developing countries.

This paper explains the background, goals, planned deliverables and progress of the \textit{MI}^2\textit{Cl} project. Section 2 gives a short characterization of the industrial development challenge. Section 3 describes the \textit{MI}^2\textit{Cl} concept, it explains an architecture of an information and command infrastructure and the role that enterprise and artifact models play in the development and use of the infrastructure services. Model execution engines and innovation coaches are two important generic services. Section 4 describes the current status of the \textit{MI}^2\textit{Cl} project.

\section{THE INDUSTRIAL DEVELOPMENT CHALLENGE}

\textit{The Development Target}

Advanced and future industries are characterized by their ability to produce a large variety of products in a lean, agile and sustainable way, and by doing this with consideration of the complete life cycles of the products and production means. For details on particular production management and engineering techniques one could check Womack \textit{et al.} (1990) for an account on the development of lean production, Goldman \textit{et al.} (1994) for a description of agile enterprises, and Alting and Jørgensen (1993) for techniques to achieve sustainable production and life cycle assessment.

Summarizing, and without paying attention to how to achieve these qualities, one can call an enterprise or industry \textit{lean} when it is capable of achieving results without using superfluous resources (e.g., equipment, workers, investments, stock), it is called \textit{agile} when it is capable of responding to change quickly and intelligently. An industry is sustainable

\footnote{\textit{UNU}/IIST: United Nations University, International Institute of Software Technology (in Macau).}

\footnote{\textit{UNIDO}: United Nations Industrial Development Organisation (HQ in Vienna, Austria).}
when products are designed, produced, distributed and disposed with minimal (or none) environmental and occupational health damages, and with minimal use and disposal of resources (materials and energy) (Alting and Jørgensen, 1993).

The Leverage of Infrastructure
Emerging industries face several difficulties in rapidly moving from an early stage of development to a mode of production in which extended enterprises (Browne et al., 1995) and virtual enterprises (Goldman et al., 1994) show flexible responsiveness, also to needs of the local market, and achieve product variety, sophistication, optimal usage of capacity, and environmental sustainability. Difficulties are caused by plenty of factors, including the absence of a favourable business environment and physical infrastructure, the lack of human resources – in a wide range of specialized skills –, the scarcity of capital, and the lack of technology. This mix of difficulties can not be overcome by a single measure. The proposers of the MI2 CI project expect a positive effect from the combination in an information and command infrastructure of consolidated ICT, insights regarding business process engineering (Hammer and Champy, 1993), and systematized knowledge about products and processes as captured in artifact and enterprise models. An information and command infrastructure may help to raise productivity and lower production costs, as do the traditional infrastructure networks, notably in the areas of sanitation, water, power, transportation, irrigation, roads and telecom. In addition it may leverage the innovation of products and processes and be an enabler for industrial development.

Failures and Successes of Past IT Deployment
In spite of the theoretical importance of information technologies “many developing countries are often suspicious of information technology as an agent of perpetual dependence upon industrialized countries, and feel threatened by informatization” (Yamakage, 1990). Also, for projects that have been implemented, the understanding of the impact of information on development has remained largely anecdotal, and evaluation of interventions has usually been related to short-term outputs (Stone, 1993).

The skepticism regarding the impact on (industrial) development of traditional information technology – i.e. “isolated” databases or software packages – may be justified: Information technology is usually provided by vendors for specific, detailed needs, and users acquire packages one-by-one. After some time, needs are encountered for exchanging data between packages, for example via underlying databases, and for functions of one package to invoke functions of another. Often such needs have been frustrated by the inability to link up such data and such functions, or by the error-proneness of such links. As a result several investments seem to miss their targets. Moreover, developing countries are often not in the position to meet the resulting perpetual demand for investments, without the return to earlier investments being proven.

Similar interfacing problems with the use of computers and software applications have appeared elsewhere. For instance in manufacturing (Van Houten, 1992). In the field of business systems the problems of interfacing between various function domains has nurtured the development of enterprise wide information systems – built around databases – which achieve an intra-enterprise information and command infrastructure functionality (Scheer, 1994). For manufacturing systems, and in various factories, in house solutions
exist (Matsuda et al., 1993).

By considering the wider scope aspects of enterprise operations - as is done in CIMOSA (AMICE, 1993) and ARIS (Scheer, 1994) - and also projects (innovations), market operations and extended-enterprise projects - as intended in the MI²CI project - one can specify and develop information and command infrastructures which provide information, decision and control support for full business processes within enterprises, extended enterprises and markets or industries as a whole.

**The Role of SMEs**
The production of large varieties of high-value products requires networks of enterprises, including many small and medium size manufacturers, to innovate and concert value-adding processes.

The important share which SMEs have in an economy stem from the variety of services and products which they supply, the low costs at which they operate, the relative ease with which they can innovate, and the number of jobs they create. Many countries have sought ways and means to stimulate and increase the numbers of small firms starting up. The possibilities for small businesses to link up to an information and command infrastructure have to be considered from the outset.

3 **MI²CI: THE CONCEPT**

A manufacturing industry information and command infrastructure (MI²CI) is a large and complex system without centralized control, which supports the cooperative behaviour of many agents (public bodies, enterprises and consumers) having their own independent interests, values and modes of operation. These agents should meet economic, social, sustainability and environmental challenges and therefore cooperate and compete, abiding by rules spelled out in the business environment.

A manufacturing industry information and command infrastructure can enable the development of virtual enterprises and joint product and process development by small and medium enterprises at much lower costs than at present. Tools for ERP (enterprise resource planning), CAD, CAM and CIM, and CALS services can be interfaced to the infrastructure, and new applications can be designed to draw on the infrastructure services. By positioning new applications or services in the infrastructure, they can be focussed, and impacts and enabling role for business and manufacturing processes can be assessed more accurately.

3.1 **The MiViPoRo Framework**
The MiViPoRo framework (short for: Modules for innovation, Versions for improvement, Proxies for operation, Records for observation) is proposed by the first author (1996) to guide the requirements definition and development of generic system services for an information infrastructure for manufacturing industry. The framework serves as a basis for unifying information and command requirements of autonomous agents as they involve in enterprises and the life spans of artifacts. It offers guidance for organizing the future development of artifact life span oriented applications for manufacturing industries, and
shows opportunities for sharing applications and information.

MiViPoRo divides the problem domain of manufacturing industry into two sub-domains and links the required generic services and primitive objects to four activity layers.

The sub-domains are: the physical domain comprising the physical space, time and matter, with artifacts, agents and cells (spatial units) having life cycles in it; and the cybernetic domain which adds communication and control services to the physical domain. In the latter domain each (physical) entity is represented by at most one proxy. The term interflow denotes the coordination and monitoring of time&space&matter situated physical processes by means of computational processes in the cybernetic domain. Interface channels exist between physical objects in the physical domain and proxies in the cybernetic domain.

The four activity layers span the two sub-domains and cover observations, operations, improvements and innovations (compare with the three layer model of Inagaki (1993) (operations, improvements and innovations)). In each of these layers, work – action in the physical domain – has to be connected to computations and communications in the cybernetic domain.

The MiViPoRo framework identifies the following generic services: innovation coaches, version managers, secure model execution engines, browsers and report generators, and user interfaces.

**Model Execution Engine**

A model execution engine is a software application that manages the use of (sections of ) enterprise models, enterprise data - including workflow data - , artifact modules and artifact data in interactions with one or more agents during their work (as part of a business process involving a number of artifacts - products and/or resources - ) – see also CEN Report 1832 (1995). Model execution engines support interfaces for all agents - employees at companies and public bodies, and consumers - involved in any of the life phases of an artifact type or occurrence, in reference to the modules of the artifact type.

The model execution services should ensure some general properties, such as: (a) access restrictions and security requirements are enforced during access to the system; (b) a minimal number of records is kept, for artifact types and occurrences such that the recollection of an artifact history or artifact model (global data sourcing) and the (global) propagation of changes is always easy (either in synchronous or asynchronous mode); (c) a wide range of functions are supported, that are typical for the agents dealing with artifact types or occurrences (e.g., producers, retailers, transporters, customs, consumers, market researchers, ...); (d) model execution engines must be capable of coordinating processes involving distributed agents.

**Innovation Coach**

An innovation coach is a software system that supports (business) engineers in: (a) designing (distributed) artifact possible life models and (virtual) enterprise models; (b) evaluating alternative designs with respect to criteria (performance, manufacturability, reliability, (life cycle) cost, safety, ...); (c) implementing (realizing) the production systems (often enterprise networks) that can source components, produce and distribute, maintain and dispose the artifacts.
Innovation coaches support innovation layer activities. ICT applications can be developed which implement standard protocols for sharing information and for coordinating decisions and control in the innovation processes of extended and virtual enterprises.

3.2 A Hypothetical Federation of ALPS

An information and command infrastructure responds to the need for efficiency in business and operations for artifact life spans. As a product or resource progresses through its life it gets involved in several possible or required situations with users, owners, traders or other specialized agents. Each of these agents has skills or interests that are typical for the life phase of the product type or of its occurrences.

One possibility is to base the MI²CI services on a federation of Artifact Life Phase Service bodies (or ALPS) as illustrated below. ALPSs use the generic services of innovation coach and secure model execution engine in interactions with agents dealing with products and resources to offer specific services. The services which different ALPS offer, should be complementary. Taken together these services should extend over all possible life phases of a wide range of artifacts, such that for each possible or required phase in the life of any artifact and for the agents involved, there are ALPSs that can provide the relevant support, at any place, whenever it is needed. Typically, agents will operate within certain territories. The global connectivity that is offered by telecommunications technologies results in less restrictions for the ALPSs. A hypothetical federation of ALPSs could comprise the following:

**Central Artifact Register (CAR):** The CAR register is used for classifying (in a universal classification) and uniquely identifying artifact & material types, naming (trademarks), coding, certifying, patenting all products, parts and goods that are produced, disposed or traded in a national or international territory. The CAR could also be used for keeping track of volumes or quantities imported, consumed, produced, exported, and disposed or recycled.

**Sectorial Artifact Data Warehouse (SADW):** The SADW is used for keeping artifact life phase models and product histories that are typical for products as they occur in an industrial sector or market segment (e.g., there could be an SADW for cars, and another one for electronic components). A supplier should place its assortment in the warehouse and be responsible for its own information. Customers can indicate for which artifacts they want to receive updates. Users should be able to indicate who may or may not receive basic product information.

**Proprietary Artifact Data Warehouse (PADW):** The PADW is used by a producer, trader, recycler or disposer to store all data relevant for any of the product or resource life phases (production, transportation, use, maintenance, repair, dis-assembly, recycling, safe disposal) in which he is involved. The life phase models and artifact histories are to be stored and made accessible to authorized users, including public authorities (e.g., for safety and environmental certification), or anyone else who may get involved in the life span of the artifact type, or one of its occurrences.

**Private Artifact History Records (PAHR):** The owner of an occurrence of a artifact type (with a certain persistence, resource contents or value) is encouraged to maintain a artifact history describing all milestones in its life (e.g., upgradings, repairs, maintenance, sellings). This artifact history could be passed on with the artifact when it changes owner.
In a country, the federation formed by one CAR, SADWs (e.g., one or more per industrial sector), PADWs (one or more per company), and PAHR’s (one per consumer) forms an instrument for artifact life cycle and business operations and process engineering, entrepreneuring and (industrial) policy planning. It could support a wide range of functions that are typical for the various agents, artifact types and artifact occurrences as they meet in possible situations.

The federation of artifact life phase services can support the global sourcing of artifact model data (types) and process models (types), artifact histories (occurrences), and workflow elements (occurrences) in support of identification, decision and action, prior to the global propagation of the consequences of these actions. A federation of ALPSs can provide us with an infrastructure on which applications can be developed for improving the productivity of global sourcing and global distribution, for environment protection, reducing the use of energy and materials, and increasing recyclability and refurbishment.

3.3 Meeting the Needs of SMEs

SMEs are weak in acquiring know how, capital, technology and human resources. An information infrastructure, conceived as a federation of artifact life phase service bodies, can meet the needs of SME’s in (at least) two ways:

(i) MI²CI facilitates the communications of an SME with its business partners: The role which an SME plays in a production process, usually concerns a specialized manufacturing process for one or a few life phases of parts. In order to deploy its capabilities – the core competences – for a range of products, the SME has to involve in business processes with a number of other market-players. The number of partners may grow with the number of end-products to which the SME contributes. Communications also increase when production becomes customer-order driven (Browne et al., 1995). For the SME it is vital that it can focus on core competences and that it can rely on standardized and secure information infrastructure services in support of its communications. The infrastructure services should extend over operations, improvements and business and engineering innovations (contract acquisition, product and process innovations, quality assurance).

(ii) MI²CI supports the easy incorporation of changes in the business environment into the SME’s processes. The business processes of SMEs are to a large extent influenced by the environment in which they are active. Likewise many commonalities between (partial) enterprise models for SME will have their origin in the market rules (the confluence between enterprise operations and market behavioural rules). For the SME it will be important that changes in the rules and conditions of the business environment get automatically, or with minimal burden, reflected in its business processes.

4 THE MI²CI PROJECT

4.1 Expected Deliverables

The value added services that a global network can offer to manufacturing industry depend on a standardized and systematized representation of the data on products, processes and manufacturing technology. Standards in enterprise modelling (ENV 40 003 (Cen/Cenelec,
1990) and ENV 12 204 (Cen, 1995) ) and product data technology (STEP) (Gielingh, 1993) are very relevant to achieve this. Also the GNOSIS project (Toyama, 1996) of the IMS Program (Intelligent Manufacturing Systems) (IMS, 1994) focusses on knowledge systematization.

The Mi2Cl project aims to accelerate the introduction of information and command infrastructures for industries in developing countries. The project proposes radically new opportunities for synergies involving industrial policy planners working at the global and national levels, entrepreneurs and engineers working at the company level, and consumers. It identifies objectives and technical deliverables (enterprise, product and process models, services) for three levels of cooperation (multi-lateral, national, and among companies and consumers) and two levels of competition (among countries and among companies) in industrial development.

Global, International (Multi-lateral) . A multi-lateral information infrastructure would include: (i) Possible lives models of artifacts, processes, plants, enterprises, market and industry which have a general value and are no longer competitive. These modules would typically be installed at CAR and at SADWs and be expressed in the prevailing international standards. (ii) Techniques, standards, and software tools for constructing and implementing new modules which are compatible with those at the CAR and SADWs (the constructed modules would incorporate product properties on which companies wish to compete in the market). (iii) The definition of a basic federation of ALPSs which should enable a country in sustaining a basic industry and infrastructure (including companies and public bodies for roads and transportations, telecom, energy, water, waste disposal, agriculture and food industry, health services, repair shops ...).

National (country-wide) : (iv) Using the techniques, standards, and software tools for constructing and implementing new modules provided at the global level, a country or region can define modules which it sees fit to its industrial development, and implement them at particular SADWs, PADWs, and PAHRs. These modules and ALPSs would be seen as extensions to those offered at the multi-lateral level.

A country-wide infrastructure, embedded in a multi-lateral infrastructure, can clearly play a role in a competition between countries. Access rights should be handled within the constraints that are agreed at the multi-lateral level.

A country-wide manufacturing industry information and command infrastructure , comprising dedicated life cycle service bodies, would support market operations and entrepreneurial projects. It would also be an instrument for education and human resource development.

Local, for companies and consumers : (v) The multi-lateral infrastructure, extended with the national information infrastructure, forms a context where companies can compete on core-competences in producing artifacts and providing services. At this level the PADWs of companies and the PAHRs of consumers are introduced. They play a role in the competition between companies, again within the constraints consolidated in multi-lateral and national infrastructure services.
The company information systems – typically designed or developed in (computer aided) entrepreneurial or engineering projects – would support the operations of the company and its interfaces to the market and industry. Its embedding in the national and multi-lateral infrastructure ensures a maximal reuse of available services and minimal overheads when responding to change in the business environment.

Software tools and the modelling framework could support the carrying out of entrepreneurial feasibility studies (as described for instance by Behrens and Hawranek (1991), in a manner which draws on concurrent engineering (see for instance Sohlenius, 1992), and the modelling approaches supporting it (Kimura, 1993).

An exploitation strategy for the multi-lateral and national information infrastructure systems should draw on a detailed study of infrastructure economics, and relevant findings from public utilities, transportation and especially telecommunications infrastructure.

4.2 Current Activities: Technology Development
The achievement of the long-term goals and deliverables of the MI²CI project is pursued along different lines. Progress on the formation of a partnership to develop and demonstrate the MI²CI functionality has been slow. At present most emphasis is on technology consolidation and development.

Enterprise and Industry Modelling
The mathematical modeling of enterprises (in all aspects of marketing, administration, finance and especially development and production), supply-chains and products, is a prerequisite for the systematic development of the model execution engines and innovation coaches which will animate the information and command infrastructure.

As regards the problem domain modelling, the project draws on the use of enterprise models and reference models as tools for organizing and integrating information about enterprise processes. This area has been well established. See for instance ENV 40 003 (1990), Scheer (1994), AMICE (1993), Spur et al. (1994). Goossenaerts & Bjørner (1994) introduce also the concept of industry model.

Model Execution Engines
At present researchers at UNU/IIST are studying the applicability of the emerging ODP (Open Distributed Processing) Reference Model as the underlying computational structure for agile manufacturing in an information infrastructure environment. Based on the formal model and using the underlying computational structure, one is approaching an implementation of the prototype model execution engine for education and training - computerised business game simulating decision-making in real-life manufacturing.

Towards this goal the usual UNU/IIST methodology is followed. The prototype development proceeds in four stages:

(i) Broad (informal) study of the issues in manufacturing industry, as they appear in the intra-enterprise, inter-enterprise and inter-market contexts. The emphasis is on decision-making within marketing, administration, finance and production activities.

(ii) Formalization of the structure and operations of a manufacturing enterprise in RSL (RAISE, 1992), and how different enterprises interact in the network of suppliers. This
will take marketing, finance and production aspects into account and ultimately provide requirements for simulation software - the business game. 

(iii) Refinement of the model above, showing how simulation software can be implemented on the ODP computational platform. We shall demonstrate refinement to preserve essential properties of the model. 

(iv) Construction, first using tools to translate RSL specifications into C, of the prototype simulation software, and using Athena libraries for windowing and graphics, and CORBA (OMG, 1991) for ODP. 

As of May 1996, the M1^2Cl project has accomplished a portion of each of the four stages. A domain analysis has been conducted on the intra-enterprise level with heavy focus on production. The domain can be expanded in two directions: vertical and horizontal extensions. The vertical extensions will deal with inter-enterprise and inter-market abstraction levels. Inter-enterprise analysis will highlight the market and the trade that goes on between enterprise in the supply chain. Inter-market will focus on the interactions between markets and the general support relations across all goods. The horizontal extension will deal with deeper analysis of the four aspects of a manufacturing enterprise. Existing mathematical and qualitative techniques and tools are to be taken into account in modelling the generic model. The RSL model of the manufacturing enterprise will follow this further development.

A business game was also developed which captures aspects of a manufacturing enterprise in a gaming environment. The game is formalized through an RSL model and later converted into a simulation software. The software uses C and Athena which allows the game to be played on a local area network.

5 SUMMARY AND FUTURE CHALLENGES

The industrial development challenge and the enabling role which information and command infrastructure may play in meeting this challenge have been sketched. SME in particular should seize the opportunities which are offered by new generic computer networking technologies. To this end a systematization (into modular models) of knowledge about business and manufacturing processes and product life cycles is required, and model execution engines and innovation coaches must be developed. The latter should be capable of dynamically binding distributed model components and occurrences in support of decision making and distributed workflow management.

This paper has explored the problem domain of advanced manufacturing and considered the difficult situation of SMEs as players in this domain. An architecture of an information and command infrastructure for manufacturing industry, and the services of model execution engines and innovation coaches, have been described.

The M1^2Cl project has been proposed through its goals and current activities. Major challenges for the future are: the demonstration of distributed artifact and process models, the development of secure model execution engines to animate them; the prototype development of an innovation coach; and the identification of a partnership which can develop and demonstrate the M1^2Cl infrastructure services.
6 REFERENCES


Environmental information systems based on physical flows

A.J.D. Lambert, M.H. Jansen, R.V. Schuwer, M.A.M. Splinter
Eindhoven University of Technology, Graduate School of Industrial Engineering and Management Science, Paviljoen, PO Box 513, NL-5600MB Eindhoven, The Netherlands; phone +3140 247 2952; fax +3140 2432 612; e-mail {m.h.jansen; a.j.d.lambert; r.v.schuwer; m.a.m.splinter}@tm.tue.nl

Abstract
This paper deals with the basic aspects of environmental information systems and advocates their integration in the information systems that already exist in companies, especially those focused on production control. A strong emphasis is on the central role of physical information, i.e. material and energy flows. A multiple-input multiple-output physical-flow model is proposed to be the standard module for describing the whole range of primary industrial processes and the basic reference to information related to production. It is concluded that the logic consequence of integrating environmental information in the existing information system results in a production oriented information infrastructure concept, in which environmental information is an integral part rather than an extension. This serves clearness and flexibility of the information system and will be profitable to the core activities of the company as well.

Keywords
Sustainable production, environmental information systems, physical flow information

1 INTRODUCTION

Growing environmental concern has encouraged consciousness on resources, waste and emissions. Concepts like sustainability and industrial ecology (Graedel and Allenby, 1995) are increasingly applied. However, also a number of leading edge manufacturers are viewing environmental improvement as a means to competitive advantage, usually as part of Total Quality Management initiatives. Benefits can be obtained from cost reductions in energy use, materials and packaging, less waste disposal and reduced regulatory burdens, but also from increased production efficiencies, marketing opportunities -green image- and balanced product design (AMR, 1994). To meet these requirements, it is indispensable to generate and exchange an ever increasing and changing amount of environmental information with respect to products and production processes. Most of this information is based on physical flows.

Existing Enterprise Resource Planning (ERP)-systems on company level are focused on production control and support operational and tactical functions like cost accounting, capacity and resource planning, stock control and quality management. Although these functions are, in principle, also related to information on physical flows, their integration with environmental functionality is not performed in practice.
1.1 Environmental, physical and production data

The discrepancy between the environmental and production point-of-view originates from history, the first minimizing environmentally harmful flows and the latter optimizing the timeliness-location-quantity relation. Additionally, a gap between flow and discrete production has been grown over the years. Both features have led to different approaches in process description. In discrete manufacturing, bills-of-material are applied which describe the composition of intermediate and final products in terms of parts and modules. In process industries, recipes are applied, which describe the ingredients and process parameters. Often a recipe admits variation and, in general, multiple inputs and outputs are present.

The latter property however, is not confined to the process industry. This can be made clear if waste and emissions of a production process are considered as -though undesired- products. In this paper, the starting point is a single view on process modeling, which is of crucial importance to close the gap. Recent examples of the existing, non-integrated, viewpoints are, e.g. (Nemerow, 1995) with respect to the process industry, (Inoue et al, 1992) with respect to discrete manufacturing and assembly and (Wang and Johnson, 1995; Wong et al, 1993; Lambert, 1995) with emphasis on product recycling.

In integrating environmental information in production control systems, one is faced with the essential role of physical quantities in virtually each functionality. Therefore, a novel approach in production systems is required, by restructuring these systems in such a way that the primary (material transforming) process and its derived physical flow model play a central role. This will be elaborated in this paper. The first step in the analysis is the translation from the stated environmental requirements into physical flows concepts; the second step covers the translation of those concepts into specifications for process registration.

1.2 Structure of the paper

The starting point of this paper -section 2- is formed by the analysis of environmental requirements (2.1) and the applied concept behind physical flows: the mass and energy balance (2.2). Their mutual relationship is made clear by an example from automotive industries, taken from (Langeveld, 1995), in section 2.3. The information requirements for environmental information systems, resulting from the mass and energy balance, are expressed in the concept of a multiple-input multiple-output process approach and further detailed in an information infrastructure -section 3-. This section is concluded with a datamodel and evaluation to cover the above mentioned concept. Finally, some conclusions are established and recommendations for further research are presented.
companies with respect to image and claims because complaints are frequently related to it. Nuisance, however, is locally experienced, not directly related to production and possesses a mainly subjective character, inflicting neither considerable nor persistent damage to the eco-system.

Utilization of exhaustible resources and discharge of matter and energy to the ecosystem are closely interconnected due to mass and energy conservation. Moreover, they are relatively unambiguously quantifiable and directly related to production. Although determination of quantitative relationships between discharge of physical flows and environmental impact still remains complex and can often be performed only ambiguously, agreements on this topic have been made and standards have been established so these will usually be accepted for validation of the environmental information that has been made available (Guinee et al., 1993ab). The very essential of environmental information, however, is related to the physical flows within and through the company.

2.2 Physical flows

Production chains comprise the trajectory from extraction (mining, agriculture) up to a desired level of production (e.g.: materials production); product chains involve the physical flow through the techno-system from cradle (extraction) to grave (discharge), thus consisting of the phases: extraction, production, consumption, upgrading, disposal. The production phase can coarsely be subdivided in about three sub-phases, often carried out by different companies, namely:

1. Materials production (process industry), the creation of internal characteristics;
2. Parts production (discrete production), the creation of external characteristics;
3. Final production (assembly), the creation of complex functionality.

Increasing importance of product recycling and the establishment of techniques like life cycle assessment (LCA) -that in the near future will be standardized into detail in the ISO 14000 series, see (ISO, 1992,1994)- results in a growing demand on environmental information that is related to the total product life-cycle and thus to the physical flows. Although LCA considers usually only partial material flows -referring to those substances that may cause environmental harm-, requirements on flexibility and generality lead to the preference of founding the environmental information system on the complete physical flow scheme on company and/or product chain level. The degree of detail of such a scheme can then more easily be adapted to a variety of requirements. Favoring appropriate data-exchange between the different links of a chain, the data should be structured in a generic way, without essential distinction between particular production types like materials processing, parts processing and assembly. The product chain actually consists of an alternation of processes and products, a so-called process-product chain (Kandelaars et al., 1996). In figure 1, a highly aggregated example of a part of such a chain is presented for the automotive industry. It is seen that both process and product level are characterized by combinations of convergence and divergence. Actually, both energy use and environmental effects caused by emissions intrinsically lead to divergence, even when
the process is strictly convergent, like in assembly. This approach is the opposite of the linear or convergent tree-like structures that are commonly applied in modeling of discrete manufacturing or assembly, and in standard LCA.

There is a direct relation between local and chain aspects: information on the physical flows is supplied by local organizations. To accomplish this, material flows are subdivided according to their composition e.g. by elements, compounds, materials, families of parts (e.g. printed circuit boards), parts or modules. In general, completeness of data on composition, like the complete subdivision in elements, is not mandatory. Only data on some critical elements (e.g. cadmium) or compounds (e.g. SO₂) are required. For logistic purposes, only the subdivision into parts as given by the BOM is needed, without data on composition. A correspondence between those viewpoints is highly preferable.

2.3 Mass and energy balances

The above mentioned view results in a conceptual model for physical flows that will be used as basis of the environmental and information model (Splinter, 1994) (figure 2). The main idea is that a production system is considered as a combination of two transformation processes. Energy carriers are transformed (E-process). The energy thus released is transferred and applied to the transformation of the materials flow, i.e. the proper production process (M-process). Two entering flows can be distinguished: M₁, entering the M-process and M₃, entering the E-process. At the output side, three leaving flows are discerned: used energy carriers (Qᵥ), transfer losses (Qᵥ), and the leaving flow M₂ that can be subdivided in a product flow and an undesired flow that may cause environmental harm. Due to mass and energy conservation, the total entering mass and energy flows equal the leaving ones, respectively. This enables the derivation of mass and energy balances.

2.4 Example from automotive industry

The car manufacturer “CAR-E” desires to obtain environmental information on the complex production process and starts with a study on its engine spray cabin (figure 3) because it is expected that this represents the essential aspects of environmentally relevant processes. The engines are subject to three main processes: cleaning, spraying, and drying.

With respect to the above defined input, the following typology of flows is applied:

1. Materials flows:
   - Proportional to production level; the relevant quantity to coating is the external surface in m² of the engine. Its relationship with the engine’s weight in kg depends on its type. The type of coating also depends on the engine’s type.
   - Independent of production but proportional to time; cleaning operations occur according to a pre-determined schedule, e.g. each two weeks, independent of the level of the production.

2. Energy flows: in this case they are essentially proportional to time.
Table 1. Mass flow proportional to production level.

<table>
<thead>
<tr>
<th>material (name)</th>
<th>input (kg/yr)</th>
<th>on products</th>
<th>discharges hazardous waste</th>
<th>to water</th>
<th>to atmosphere solid</th>
<th>to atmosphere volatile</th>
</tr>
</thead>
<tbody>
<tr>
<td>benzine</td>
<td>9600</td>
<td>0</td>
<td>9600</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>primer</td>
<td>911</td>
<td>216</td>
<td>94</td>
<td>6</td>
<td>3</td>
<td>592</td>
</tr>
<tr>
<td>-base</td>
<td>481</td>
<td>228</td>
<td>99</td>
<td>7</td>
<td>3</td>
<td>144</td>
</tr>
<tr>
<td>polish</td>
<td>23761</td>
<td>10020</td>
<td>4341</td>
<td>295</td>
<td>148</td>
<td>8957</td>
</tr>
<tr>
<td>-base</td>
<td>1200</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>-thinner</td>
<td>2045</td>
<td>971</td>
<td>419</td>
<td>29</td>
<td>14</td>
<td>612</td>
</tr>
<tr>
<td>total</td>
<td>37998</td>
<td>11435</td>
<td>4953</td>
<td>337</td>
<td>168</td>
<td>21105</td>
</tr>
</tbody>
</table>

Table 1 shows the mass balance of the materials flows that are proportional to the production level, except the flows of air, water, and products. It depends on production level and product mix, resulting in a total surface (in m²) and a distribution in types of primer. A typical case, integrated over a year, has been presented here.
Table 2 presents the materials balance of the material flows that do not depend on production. They emerge at periodical cleaning programs, e.g. a monthly and a yearly one.

**Table 2.** Mass flow independent of the production level.

<table>
<thead>
<tr>
<th>material (name)</th>
<th>input (kg/yr)</th>
<th>on products</th>
<th>discharges</th>
<th>hazardous to water</th>
<th>to water</th>
<th>to atmosphere</th>
</tr>
</thead>
<tbody>
<tr>
<td>thinner</td>
<td>145</td>
<td>-</td>
<td>130</td>
<td>-</td>
<td>-</td>
<td>15</td>
</tr>
<tr>
<td>stripping fluid</td>
<td>240</td>
<td>-</td>
<td>144</td>
<td>-</td>
<td>-</td>
<td>96</td>
</tr>
<tr>
<td>rinsing fluid</td>
<td>1170</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1170</td>
</tr>
<tr>
<td>total</td>
<td>1555</td>
<td>-</td>
<td>274</td>
<td>-</td>
<td>-</td>
<td>1281</td>
</tr>
</tbody>
</table>

Finally, the energy consumption is presented in table 3. In this particular case the energy flows are basically not related to production but depend linearly on production time, except the polish mixers, that are functioning continuously.

From this case, it became evident that mass balances play a valuable role in estimating the environmental impact of production. Complete and actual information to the grade of detail presented above, however, is often hard to obtain. It should be consciously analyzed to which extent aggregation of information may be applied. Additionally, an information system can be of help if it covers both environmental and other material related issues, e.g. purchasing, scheduling, etc.

**Table 3.** Energy consumption.

<table>
<thead>
<tr>
<th>process (name)</th>
<th>input (GJ/yr)</th>
<th>output</th>
<th>heat to air</th>
<th>heat to water</th>
</tr>
</thead>
<tbody>
<tr>
<td>additional ventilation for cleaning</td>
<td>58</td>
<td>58</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>spraying and drying</td>
<td>94</td>
<td>94</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>water-curtain</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>10</td>
</tr>
<tr>
<td>polish mixer</td>
<td>12</td>
<td>12</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>compressor</td>
<td>17</td>
<td>-</td>
<td>-</td>
<td>17</td>
</tr>
<tr>
<td>lighting</td>
<td>89</td>
<td>89</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>total</td>
<td>280</td>
<td>253</td>
<td>27</td>
<td></td>
</tr>
</tbody>
</table>
3. THE TRANSLATION OF THE PHYSICAL FLOW CONCEPT INTO AN INFORMATION ARCHITECTURE

In this section, the physical flow concept of mass and energy balances is considered from an information point-of-view. The conceptual model is based on the multi-input multi-output process approach and results in an information architecture, illustrated by a datamodel.

3.1 Communication view

Environmental information is required both within the company on different hierarchical levels and by a variety of external stakeholders who act autonomously, so environmental information supports most different needs and interests. Because the field of environmental data requirement is rapidly evolving, environmental information systems should be flexible within a high degree to meet future needs. It is apparent that the manufacturer is benefitted by continuous, and if possible, growing sales, within the constraints of regulations and maintaining a solid and reliable image. From an integral chain control point-of-view, however, constraints will be imposed by suppliers and buyers to prevent this kind of sub-optimalisation. Additionally, those parties in the supply chain will request assurances to secure their position, based on a type of information not necessarily useful to the manufacturer himself. This tension is even more explicit in relation to the government, neighbours and environmental bodies. In this paper, we focus on the manufacturer, in relationship to the parties mentioned above.

The structure of the system and its interface with the users should be according to ISO 14000 standards (ISO, 1992; ISO, 1994) that are presently worked out and will cover the topics: environmental management systems, -auditing, -labelling, -performance standards, harmonisation of life cycle assessment, and guidance for product standards.

3.2 Multi-input multi-output process approach

Physical flows are considered as a very elementary and flexible starting point. However, physical data as such cannot be regarded as environmental information. The transformation of physical data, e.g. mass balances, into environmental data can be accomplished by applying the following structure:

1. Basic registration: storage of data resulting from measurements, standards, simulations etc;
2. Interfaces with specific (environmental) applications;

The tuning of environmental data with production data is achieved in the basic registration, the part that is depicted in this paper. The frequently promoted integration of environmental systems with quality data, health and safety systems etc, fits in this view on level 3. This can only be realised if the underlying data are available and consistent, and provides that a consistent view on the basic registration is available.

Two approaches to information that play a crucial role are discernable:
1. Information of mainly qualitative character like procedures and instructions that are part of the environmental management system. This information is particularly top-down oriented (from management towards primary process) and can be handled within a workflow management environment.
Information of principally quantitative nature, reflecting the physical situation of the production system and possessing a bottom-up orientation, thus directed from the primary process towards the management. This study is mainly focused on the latter approach.

Because environmental management systems have a similar structure as other, e.g., quality and safety, management systems, it is highly preferable to combine them. This principle should also govern the structure of information systems. A scheme of the information flows that are related to environmental topics is expressed in figure 4.

Central in this picture is the primary process, represented as a multiple-input multiple-output process with its multiple entering and leaving physical flows. Parallel to it, a physical flow model is present that meets the requirements of all users of physical information. It is based on standards and it is characterized by internal degrees of freedom that should be adjusted to the reality from time to time. To that purpose, data are transferred to the physical flow-model that may be the results of measurements, data on stocks, production, routing and so on. Here it becomes clear that also data from non-environmental management systems like production control are transferred to the physical flow model. Within this approach, the environmental management system exhibits a generic structure in common with other management systems and is actually fully integrated with them. As is visualized in fig. 4, the environmental management system generates information for internal use, e.g. for supporting cost accounting, and it may also support chain information systems by exchanging information from and to other links in the chain. Moreover, it should provide information for external use, varying from public annual reports to data in favor of agencies that grant licenses.

Figure 4. Physical information flows.

3.3 Information architecture

In this section, the outline for an information (reference) architecture is depicted for the basic registration of an aspect system, in casu an environmental information system. Earlier, it is advocated that an aspect system should be developed as an integral part of the entire enterprise resource planning (ERP) system and thus a number of areas of a customer-oriented organization are involved. (AMR, 1994) summarizes the following functional areas:
• **research and development**: the design stage in a product life-cycle approach commits to most of the costs and problems.

• **production logistics**: this includes (1) materials management, purchasing systems need guidelines and specific data about hazardous materials as a starting point for negotiations with suppliers and to be able to supply data to manufacturing; (2) scheduling and planning: just as a schedule can optimize for bottleneck equipment, it can optimize for minimal pollution; and (3) processing: many companies create reams of process data, nevertheless they cannot stay within their permitted limits permanently. It’s not that they don’t have the real-time process data they need, they simply don’t compile or save the data in a way to turn it into useful information.

• **maintenance**: the way equipment is running can have a dramatic effect on waste streams. This both holds true for production means and for durable equipment such as airco’s being under service by the original equipment manufacturer.

This view is described as the subsystem dimension. A second dimension in ERP-systems is formed by the aspect system, in this case the environmental system. An overview of dimensions is specified below. General for ERP-systems are:

• **subsystem**: the subsystems are defined as functional areas along the supply chain: engineering, purchasing, manufacturing / processing, warehousing, distribution, sales, service, recycling, ...

• **aspect system**: an aspect system is defined as that part of the organization (system, ...) belonging to the specified aspect, across all subsystems. The following aspects are distinguished: logistics, quality, health, safety, legislation, economics, ..... 

• **branch**: the type of product highly influences the type of possible environmental burden. Important categories are: chemical, food and beverage, automotive, electronics...

• **control level**: the level of control influences the level of detail required. The classification originates from general enterprise control: strategic, tactical, operational and instrumental.

• **implementation level**: state independent data: norm values (BOM/formula) or norm values (routing) and state dependent data: measurements and simulations.

Environmentally specific orientation:

• **environmental function**: there are two main differences: logistic (l) versus registration (r) aspects, and quantitative (qn) versus qualitative (ql) data. We distinguish product waste (both l-qn and r-qn), emissions (r-qn), hazardous materials (both l-qn and r-qn), packaging (both l-qn and r-qn), nuisance (r-ql).

• **environmental application**: environmental annual report, environmental management / assurance system, permits, measurements, registrations...

• **environmental solution**: materials and energy balance, environmental costing tracking and tracing and reverse logistics.

### 3.4 Towards an environmental datamodel

As was stated in the conclusion of section 3, a supporting information system is helpful to realize the implementation of mass balances. In this section the datamodel for CAR-E is given, based on the mass balance model described earlier.

At first instance the system is meant to be a prototype for internal use and therefore no attention is paid to the communication infrastructure and integration with other aspect systems. The system infrastructure focuses in particular on the basic registration based on mass balances, and includes the operational and instrumental level of material management, scheduling, planning, manufacturing and maintenance. In figure 5 the main entities of the datamodel and its relationships are shown.
The different flows as were distinguished in figure 2 can be found in this model. The effects on the environment are modeled as dependent only from the production system. There is no difference between the situation in which the environmental burden is desired or not. There are cases however where not all effects on the environment have to be accounted for. When for instance a by-product is one of the results of the production system, then following the model in figure 5 this is con-

![Figure 5. Datamodel of the environmental information system for CAR-E.](image)

sidered as a burden for the environment. It is not counted as such however when e.g. another firm can use it for its production system. The datamodel in figure 6 is more closely to this situation. There a created product can be a principal product, a desired by-product or an undesired by-product. The environmental burden of the model in figure 5 is connected to the undesired produced by-product.

![Figure 6. Proposed datamodel for an environmental information system.](image)
4 CONCLUSIONS

As a preparation to apply environmental functionality to management information systems, the essentials of this functionality have been analyzed. A top-down and a bottom-up approach are discerned here, referring to respectively workflow management and processing of information from the primary process.

It appeared that the essentials of the latter aspect are based on physical flows and a general description of production processes by a multiple-input multiple-output approach. Although not explicitly discernible within the usual structure of production control systems, virtually all information on production is based on physical flows. According to this observation, a novel approach to structuring information systems is proposed that justifies the position of physical flows in the information system and that leads to similarity on structure for the information (sub)systems. It implies that environmental information systems should not be added as a separate module, but should be integrated in the present modules, especially the operational - logistic - information, by adding new attributes to existing entities. To that order, the structure of the logistics information system should be modified to by-pass the artificial difference between discrete manufacturing and process industry by universal application of multiple-input multiple-output models in all types of production. Once performed this restructuring, derivation of environmental information has become a standard task.

5 REFERENCES


6 BIOGRAPHY

*Fred Lambert* is assistant professor at the Eindhoven University of Technology, section Energy and Environment of the faculty Technology Management. His research interest is in the field of chain management in relation to energy and the environment.

*Monique Jansen* received her masters degree in Computer Science in 1993. Currently, she is a Ph.D. candidate at the Eindhoven University of Technology in the field of environmental information systems from a business point-of-view. In this research she is cooperating with Baan Company, a manufacturer of Enterprise Information Systems.

*Robert Schuwer* is assistant professor at the Eindhoven University of Technology, section Information and Technology of the faculty Technology Management. He finished his Ph.D.-thesis in 1993 on a subject in the area of knowledge base systems. His current research interest includes IT-based solutions for knowledge management, intelligent production planning and support of professionals with advanced IT.

*Maarten Splinter* is assistant professor at the Eindhoven University of Technology, section Energy and Environment of the faculty Technology Management. His research interest is in the field of energy systems.
Strategies to develop a telemonitoring technology for machine tools via the World Wide Web

I. Goncharenko, K. Mori, and N. Kasashima
Mechanical Engineering Laboratory, Agency of Industrial Science and Technology, Ministry of International Trade and Industry
1-2 Namiki, Tsukuba, Ibaraki, 305 Japan, tel: +81-298-58-7074, fax: +81-298-58-7201, E-mail: mori@mel.go.jp

Abstract
The World Wide Web (WWW) can provide machining systems with remote monitoring capabilities through highly graphical user interfaces. As part of a feasibility study, an experimental system to check the availability of the current WWW protocols and languages for on-line machining data transfer and representation was developed. The system consists of a machine tool with PC-based NC, sensors, servers and terminal clients. The primary limitations of standard WWW techniques applied to the dynamic monitoring were found. A functionally distributed monitoring architecture to overcome these disadvantages is proposed. The present approach also provides recommendations for reconstructing complex user interfaces and virtual reality on the client side by utilizing a few of the machine tool parameters.

Keywords
monitoring, machine tool, World Wide Web, graphical interface, virtual reality

1 INTRODUCTION

Modern manufacturing with geographically distributed systems may be remotely monitored and controlled. Such monitoring includes information collection and transfer from remote machine tools to control centers, data analysis both in automated and interactive modes. The information will enable the prediction of failures in vital parts of
the processes and, consequently, reduce the risk of high-cost losses and maintain overall system efficiently. Local area networks with information transmission speed of more than 300 kB/sec are being introduced in modern manufacturing plants. These environments provide manufacturers with an opportunity to utilize the on-line data obtained from their domestic and overseas manufacturing systems through the Internet.

The Internet has been developed as one of the most important world-wide computer communication mechanisms. In the last few years, WWW, initially designed by Berners-Lee (1992) and then supported by National Center for Supercomputing Applications (NCSA), has became an essential part of the Internet. It is a distributed information storage system, which provides graphical access to users through terminals (clients) to data sources (servers). The data described as documents in terms of the Hyper Text Markup Language (HTML), can contain references to other distributed documents by the Uniform Resource Locator (URL) addressing scheme. In addition to HTML, WWW is based on the HyperText Transfer Protocol (HTTP) providing an application level of the information transmission on the Web. At present, the WWW clients have ability to explore complex 3-dimensional scenes represented in the Virtual Reality Modeling Language (VRML) documents on servers (Pesce, 1995). The dynamic content of user graphical interfaces, corresponding to real time events, is yielded by either server-side CGI (Common Gateway Interface) techniques or client-side scripting applications, e.g., the Java applets (Ritchy, 1995).

The rapidly evolving WWW now offers interactive and automated on-line monitoring, data analysis, and control of remote machine tools. However, some limitations exist in the present WWW for these applications, and require improvements and new developments in protocols, languages, interfaces and techniques for the WWW. Machine tool monitoring also demands a specific system architecture.

In this paper we discuss the restrictions of the commonly used WWW techniques in remote machine tool applications and control by constructing an experimental system. A distributed monitoring architecture is proposed to overcome the restrictions of extensive CGI methods and minimize the total amount of information transferred. Using this architecture, the HTML and VRML documents are reconstructed on the client side from a few on-line machine tool parameters. Consideration is also given to the WWW graphical representation of the parameters.

2 EXPERIMENTAL SYSTEM

In order to investigate the feasibility of the current WWW technologies for on-line machine data transfer, representation and control, a monitoring server-side system as shown in Figure 1 was developed.

The server-side system consists of Sakazaki SEC AE-61 3-axis milling machine controlled by Pentium PC/60 MHz computer with ST-8000 numerical controller (NC) from Techno Corp., Japan. The PC-based NC has its own RAM for CNC programs and serves the following functions; loading up to 12 CNC programs from PC disk drive into RAM and running one of them autonomously, directly controlling the machine tool, and
receiving a machine status. The status includes the real-time information about such parameters as the number of the running program, program step number, designated and actual x-,y-,z-coordinates of the tool, feed rate, spindle speed, axis motion status, and alarm time interval.

Figure 1  Experimental system architecture.

The same PC receives video information from a CCD camera through the Intel Smart Video Recorder capture board (VB). The video board delivers luminance and chrominance data and forms color images in four sizes from 160x120 to 640x480 pixels. We selected the resolution of 320 by 240 to save space in the client display frame for other textual and graphical information. The 3Com EtherLink network adapter assures fast communication in our local area net between the monitoring server and clients.

The PC-based hardware was originally used for other purposes and determined the basic software components of the system. The capture board is controlled by Video for Windows with Indeo Video drivers, and access to the numerical controller is handled through the library of Microsoft C functions developed by Techno Corp. Thus Windows 3.1 was used as an operating system with Windows-based network software.

The video capturing program can store continuous video clips, but it exploits CPU resources exclusively and requires a huge amount of disk space. To avoid these factors and automate the capturing process, it was run in frame-by-frame mode under Windows Macrorecorder. A cycled macro, emulating fast key strokes, invokes video capturing and then our self-made application to provide a current video image.

Two applications serve the NC board. The first one provides an interface between server-side CGI programs and NC, receiving current machine tool parameters and sending control commands. The second autonomous application controls the machine tool in accordance with a pre-defined CNC program. One of the purposes of the CGI applications is to transform current on-line data and image into the documents in a WWW format and send them to client programs (browsers). These dynamic, "on-the-fly" documents contain random numbers in their URL names. This forces a browser to load the documents and
images every time. Otherwise, the documents with the previously executed URLs are
extracted by a browser from its local cache.

From a client point of view, the system provides three functions at present:

- continuous graphical monitoring of the machine tool state;
- remote control of the machine tool;
- monitoring in virtual reality form.

Continuous monitoring means that a user can observe the current machine tool real image
and parameters in a textual/graphical form. The user graphical interface is updated in time
automatically. Otherwise, the remote control interface implies user interactive inputs by
clicking a mouse to move or stop the machine tool. In the virtual reality interface the user
can inspect "frozen", static artificial three-dimensional scenes corresponding to the real
machine tool position at the moment of the interface invocation. Mac, PC, and Silicon
Graphics Indigo² browsers were tested as clients of the server-side system, and the last
one showed high capabilities of visualizing the virtual reality.

3 ON-LINE DATA REPRESENTATION ON THE WEB AND DATA
SIZES

Information about the current state of machining processes can be obtained through the
Numerical Controller and various sensors. Their outputs are: tool coordinates, feed rate,
cutting force, spindle rotation speed, vibration level, acoustic emission, temperature,
optical images, etc. All these data are essentially continuous and measured in real-time, t.
After digitizing they can be depicted as a finite set of discrete one-dimensional functions of
t or multi-dimensional functions of t and other parameters (coordinates, frequencies).

But how can the data be represented on the WWW? It depends on a successfully
designed intuitive user interface, because WWW browsers only graphically interpret
server-side HTMLs/VRMLs. Originally, the Web was created to retrieve technical
documents. Thus, the simplest way is to represent the on-line data in the form of plain text,
formatted text (hypertext) and tables, utilizing current time and date together with
measured parameters transferred from a server.

Compared to tables, graphs are more human-friendly for presenting time-dependent
functions. Developing WWW for increasing the graphical capacities, NCSA added in-line
images into WWW documents in several graphical formats. One of the formats, GIF
(Graphics Interchange Format) is well suited to the representation of plots, graphs and
pictorials. GIF is based upon the lossless compression algorithm (Ziv, Lempel, 1978)
designed especially for raw data with repeated substrings such as raster images of plots.
Otherwise, the JPEG (Joint Photographic Expert Group) format with "lossy" compression
algorithm provides much better results for natural true-color and gray-scale images
(Aravind et al, 1989). It is important to choose a specific compression method for reducing
the transferred image data size without serious losses in quality of "visibility".
Figure 2 shows the user interface for remote monitoring and control of the experimental system. The interface contains on-line parameters in tabular form (1.6 kB of HTML document), JPEG video image of the machine tool (8.5 kB), GIF image of spindle speed plot (1.6 kB), and schematic GIF image of the tool position for point-and-click control (2.8 kB).

**Figure 2** Graphical user interface (remote control).

The GIF compression ratio indicates complexity of an image and depends on its nature - texture, repetitiveness of pixels, level of details and noise. Trying to reduce data sizes of photographic images, several image sequences of typical operating scenes were tested. It
was found that the JPEG pictures have 7-10 times less size than comparable GIF images, as shown in Figure 3. The lower threshold of JPEG quality factor was estimated subjectively when independent experimenters detected the difference between original and compressed images. The figure also displays the efficiency of JPEG format for color images. When JPEG compression is applied to artificial graphs and plots, it proves inefficient and causes an erosion of drawn objects.

**Figure 3** Averaged JPEG/GIF compression efficiency for typical operating scenes.

In the system, the process of monitoring enables a remote operator to observe the machine tool state and control it through an "updating-in-time" graphical interface. The graphical information is arranged in a display frame as shown in Figure 2. At a defined interval of the updating time, for example 1 sec, total data size can be estimated from Table 1. The above interface without JPEG image requires 6 kB to be transferred.

Besides JPEGs and GIFs, other WWW data with certain MIME (Multipurpose Internet Mail Extensions) media types (Borenstain, Freed, 1989) can be retrieved by the client. In our case, even simple VRML 1.0 document describing one static 3D operating scene requires about 5 kB. The size of VRML documents strongly depends on the degree of comprehensibility of the scene described.
Therefore, to represent in graphical form a few on-line parameters initially stored in 20-30 bytes on the server side, it is required to transfer to the client 5-10 kB of documents! In this case, the bandwidth of the Internet is a serious restriction for monitoring. For example, the bandwidth of our local area network varies from less than 1 to more than 300 kB/sec, depending on a server/client location and time of day. For some clients the latency between real event in the machine tool site and the appearance of corresponding graphical representation exceeds 10 seconds. This latency is too large to control the machine tool adequately or stop it in case of emergency.

Table 1  Types of on-line data and data sizes (* - 1sec playback)

<table>
<thead>
<tr>
<th>Type</th>
<th>MIME media type</th>
<th>Data size, kB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text, tables</td>
<td>text/plain/html</td>
<td>0.5..5</td>
</tr>
<tr>
<td>Graphs, plots, pictorials</td>
<td>image/gif</td>
<td>2..5</td>
</tr>
<tr>
<td>Images</td>
<td>image/jpeg</td>
<td>5..20</td>
</tr>
<tr>
<td>Audio</td>
<td>audio/basic/wav (*)</td>
<td>5..10</td>
</tr>
<tr>
<td>Video</td>
<td>video/mpeg/msvideo (*)</td>
<td>50..1000</td>
</tr>
<tr>
<td>VRML data</td>
<td>x-world/x-vrm</td>
<td>5..500</td>
</tr>
</tbody>
</table>

If it is assumed that the frame is accompanied by real-time audio/video information transferred through the same network, the data size per 1 sec increases dramatically (Table 1). At present popular browsers on various platforms include internal tools to playback audio/video multimedia files in "real-time" simultaneously with receiving the files. On the server side, the files must be prepared and described in HTML in advance, so there is not a continuous stream of information required for monitoring. Otherwise, InPerson videoconferencing technology, developed by SGI and promoted through NetManage for PCs, provides useful properties for inclusion in our system. The estimated Internet bandwidth required in both cases is at least 50 kB/sec.

Thus, the first strategy for successful "real-time" monitoring is to reduce the total amount of data transferred through the network. The limitation in bandwidth is the first restriction on monitoring via the WWW, but not the last. In our fast local network with a bandwidth of more than 300 kB/sec, three types of user interfaces were tested. With the first, including only on-line parameters in plain text format, it was possible to update the information 10 times per second. When two GIF images and then a JPEG image as shown in Figure 2 were sequentially added, the updating rate was reduced to 2 and 4 seconds respectively. Moreover, in all cases the updating rate was not time-uniform. The reasons of these disadvantages are considered next.

4 DISADVANTAGES OF SERVER-SIDE SCRIPTING

Time non-uniformity of the monitoring

Every new cycle of monitoring requires generation of the dynamic WWW documents corresponding to current operating parameters. The client-side control also is possible by
transferring the user inputs to controlling programs existing in the server-side system. As a WWW convention, these properties are maintained by the NCSA CGI technique.

CGI is an interface between client and server for running external programs, or gateways, under an information server. The gateways can be ordinary executable programs that provide standard output in the form of scripts. These scripts contain at least the MIME type definition of transferred data and the WWW document itself or its location. Commonly, input information for the gateways is passed in an interactive mode; a user of the client browser fills and submits FORM and ISINDEX requests of his current HTML document. The requests are passed through CGI and handled by a server. Started on the monitoring server, the gateways receive and process on-line data, control the machine tool, create new WWW documents and then output the appropriate scripts for the server. In such a manner several authors developed robot systems teleoperated through the WWW, for example, Goldberg of the University of Southern California (Goldberg et al, 1995).

This method is satisfactory for the interactive remote control and generation of current virtual reality scene, but the continuous telemonitoring requires the documents to be updated automatically on a periodic basis. We tested two complementary WWW mechanisms providing the automatic document loading, namely, "server push" and "client pull". In Figure 4 both techniques are shown schematically.

Figure 4 CGI techniques for continuous monitoring.

In the "client pull" method, the server sends down a special directive followed by main document body, for example;

```html
<META HTTP-EQUIV=REFRESH CONTENT= "5; URL= http://host/NewRandom URL">
```

Document body
The browser will fully display this document, then wait 5 seconds and call a new gateway defined by the URL. In this technique, a client-server HTTP connection is held open only for transferring a current portion of data. Otherwise, in the "server push" method the connection remains open for an indefinite time period (Figure 4). The data transfer is fully controlled by the server. Initially, the server sends a special MIME content type descriptor and boundary terminator, informing the browser when every portion of data will be finished. Then CGI scripts force the server to send current documents, HTML0, HTML1, HTML2 at times $t_0$, $t_1$, $t_2$ respectively. When the browser receives the terminator it waits for the next document.

Various experiments were carried out with the experimental system, and it was found that these server-side scripting techniques have the disadvantage of highly non-uniform playback on client side. In case of "server push", it is possible to tune the time difference $t_1-t_0= t_2-t_1 = \ldots$ to a constant value from server side only, but the browsers do not synchronize multipart MIME data. In "client pull", the time uniformity also cannot be defined by the META tag because a different time required for preparation of dynamic documents and images on the server. In addition, a data transfer time is unpredictable. For example, the updating time of the user interface, shown in Figure 2, was varied from 3 to 8 seconds.

Therefore, the next strategy for telemonitoring is to use a synchronizing mechanism on the client side, which will be discussed in Chapter 6.

**Extensive nature of gateways**

CGI applications use much processing time, preparing dynamic documents in WWW format. The graphical representation of on-line information adds to CGIs functions such as drawing and GIF compression. The gateways also generate random URL names for each new document. The CGI scripting describes that the gateways write new HTML/VRML documents directly on standard output or prepare them using pattern files.

When several clients monitor the machine tool simultaneously, the number of gateways increases and it can overload the server. In order to discharge the monitoring computer, the functions of dynamic document preparation should be passed from server to client side.

**5 CLIENT-SIDE DYNAMIC INTERFACE CONTROL AND VRML**

Actually, the above mentioned strategies of the telemonitoring are interrelated. A client-side control may result in the system flexibility and minimal data transfer. The successful solution of building a monitoring system is based on optimal distribution of server-client functions. We propose such distributed system architecture (Figure 5).

The server-side computer serves the functions of the machine tool control, on-line data receiving and transfer to clients in a compact file format. If the Internet bandwidth is adequate, teleconferencing can be added.

All static parts (HTML patterns, decorating images) of the graphical interface associated with monitored operating parameters are pre-loaded from an initial description of the machine tool (MTID) to the client computer. MTID contains algorithms to create the
dynamic interface from these static parts. Java portable code is used to yield a platform independence of the algorithms. On the client computer, the Java code receives information from the machine tool through its Network Class Library (Ritchy, 1995) and continuously updates the user interface. This Java code controls the time-uniformity of interface updating.

Figure 5  Distributed server-client architecture.

MTID additionally includes VRML geometrical descriptions of machine parts, CNC programs of the machine tool and NC-control algorithms both to simulate machining processes in off-line mode and calculate on-line VRML scenes. At present, in the same manner we utilize on-line data with the CGI method to reconstruct virtual scenes. For example, in the server, three parts of the machine tool, corresponding to each coordinate axis, are described geometrically in VRML as follows:

```
#VRML V1.0 ascii
Separator {
  ... ..... 
  Separator { #part1
```
Receiving the current on-line x-,y-,z-coordinates of the machine tool and CNC program step number, gateway calculates new x1, y1,..., w3 for the pattern and replaces them. Then the modified VRML is transferred to the client browser.

For the new architecture, there are no reasons to pass whole VRML documents every time, because the pre-loaded pattern is modified on the client side in accordance with the on-line coordinates transferred. It is possible to use this advantage by reconstructing VRML display images into user interfaces instead of real video images of the machine tool. Moreover, the initial VRML pattern can provide more level of details then the real image.

At present our VRML scenes are static, so users only examine a "frozen world" of the machine tool. In future applications, Cosmo tools from Silicon Graphics for VRML 2.0 will be used. This software provides moving virtual reality by Java scripting technique, which is well suited for our distributed system.

6 SUMMARY AND CONCLUSION

Future manufacturing system will need network communication between world-wide distributed manufacturing machines and information processing subsystems. This communication must serve the dual function of telemonitoring and adequate machine tool control according to the monitored information. At present, the World Wide Web as an information system on the Internet, has two remarkable features: (1) it provides an access to distributed and cross-linked data sources, and (2) it gives a graphical user interface. Our application field confirms that WWW technologies have evolved in the right direction.

With the experimental system developed, the possibility of transferring and representing information about a current machining state has been tested. The experiments confirmed that dynamic on-line textual, graphical and visual information can be retrieved on a continuos basis by using gateways, "server push" and "client pull" mechanisms. This standard CGI technique was found to be adequate for remote control. However, for continuous monitoring it has several disadvantages; consuming vast server resources, increasing the volume of non-relevant information transferred, and not providing time-uniformity of updating the user interface.

It was also found that the low bandwidth of the Internet is a basic restriction for monitoring applications. With the framework of CGI, successful monitoring is possible only for minimal graphical interfaces. Real time audio and video information requires at
least 30-50 kЂ/s of the bandwidth. Additionally, such synchronizing tools are required for "live" data streaming, as a real time protocol or InPerson teleconferencing.

To overcome these limitations, a functionally distributed architecture of the telemonitoring has been proposed. This approach reduces the total amount of transferred data and retains the rich graphical representation of machine tool operating state. It is achieved by utilizing only a few of the on-line parameters to generate a display frame (including artificial images and moving virtual reality) on the client computer.

The main results of this study for the purposes of on-line machine tool telemonitoring and control are as follows.

(1) After testing several graphical interfaces, the main restrictions of the telemonitoring via WWW were found. All the restrictions originate from the extensive nature of the standard method for generating dynamic documents. Using this method only the simplest user interfaces are available for continuous monitoring, but remote control is still possible.

(2) A distributed client-server architecture minimizing the amount of transferred data and concentrating only on on-line data was proposed. The new approach can get around the limitations by the use of the latest WWW tools.

7 ACKNOWLEDGMENT

The authors wish to thank Dr. T. Kojima of the Mechanical Engineering Laboratory for initiating our interest in application of the VRML for the machine tool monitoring.

8 REFERENCES


Towards Intelligent Manufacturing Systems
Abstract
This paper describes the Globeman project, its background and history, its vision, objectives, strategies, organization, work plans and consortium partners. The results are briefly categorized. Industrial scenarios and demonstrators are already available. These serve as specifications, guidelines and training simulators for other partners and projects facilitating technology transfer. Globeman will run through 1999 and the main industrial results and achievements will be published from mid-1997 onwards.

Keywords
IMS, Intelligent Manufacturing System, Globeman, Product Life Cycle, Enterprise Integration, Global Manufacturing, Virtual Enterprise

1 INTRODUCTION

Globeman is a project that operates under the international IMS program. It is a development from the Test Case of the same nature, that has operated since 1993 in the IMS feasibility study. Many of the partners in the present Consortium are continuing from the earlier project. The Globeman Test Case studied the requirements for manufacturing enterprises to remain competitive in the face of an increasingly demanding and selective marketplace and with rapidly advancing telecommunications and transport technologies. The Test Case took into account the way markets are steadily becoming global, rather than domestic.
The Test Case recognized, as many other studies have, that the challenges is to continually improve performance in terms of improved products, reduced costs, decreased product development time, increased flexibility to tailor products to customer needs, and the provision of cleaner products using cleaner processes. The important findings by the Consortium were how the changes in the marketplace and the drive by the new technologies can be used as opportunities for major shifts in the way manufacturing businesses are organized and managed. Three essential features for manufacturing in the future were identified. Firstly, the formation of “Virtual Enterprises”, or close collaborations between companies and organizations (usually not bound together by long-term formal arrangements) all over the world, to provide a specific product or outcome. Secondly, the need to integrate the whole life cycle of a product from initial conception through to final disposal was seen as critical, if the needs of the customer and the community are to be fully satisfied. Thirdly, the adoption of Information Technologies in ways that integrate global enterprises, to achieve really close relationships between widely scattered groups and people, was seen as an essential element for the future. Combining all of these findings from the Feasibility Study, Globeman21 is directed at achieving major improvements in the Global Manufacturing Process.

2 BACKGROUND AND HISTORY

We live in a rapidly changing world. International relationships are in continual flux. Manufacturing technologies, business practices and market demands are under constant review. Information technology, is developing at an accelerating rate. Many of the global developments have left business practitioners puzzled and confused about how best to achieve commercial advantage. This is the challenge on which Globeman 21 was founded.

Fig.1 shows the history of Globeman 21. In 1993, a International Test Case was conducted by the IMS feasibility study. We learned a lot from this test case. This test case tested advantages and difficulties in cooperation coming from multi-region/multi-national, various industrial sectors, various size of company, diversified interests by partners, different funding system, distributed R&D system and IPR(Intellectual Property Rights). After that some partners continued Transition Period Project and prepared full scale proposal. In September 1995, our proposal was endorsed by ISC2(2nd International IMS Steering Committee). Kick-off meeting of Globeman21 was held in Sydney, Australia in March, 1996.
3 PROJECT OVERVIEW

3.1 Vision

The Future will see Manufacturing Globally Integrated across Time and Space.
Globeman21 is an international research consortium building a cooperative team of the world’s best industrial and research organizations to develop and sustain collaborative networks. It is drawing on the specialized strengths and knowledge available in many organizations, to build the business practices, the management techniques, the information infrastructure, the simulation systems and the modeling tools for integrating the elements of an enterprise across geographic, cultural and time barriers.

3.2 Objectives

Objectives of Globeman project are:

1. To create the business processes; the methods, models and technologies, for the emerging global manufacturing environment. Global life-cycle management will be included as a key element. The new approaches, models, methods and technologies will be integrated by managing the processes in global enterprises.

2. To improve the quality and professionalism of manufacturing by performing several industrial Demonstrators.

3. To present clearly the findings of GM 21 so that the participants and other companies can radically improve their business processes and environments based on new business models and supporting tools.
3.3 Strategy

To achieve Globeman objectives, we concentrate on 2 business processes and 4 technology areas.

**Business Process Groups (PGs):**
- Global Product Life Cycle Management
- Global Manufacturing Management

**Technology Groups (TGs):**
- Modeling Technologies
- Technologies for Information Access and Control
- Technologies for Scheduling and Coordination
- Technologies for Business Process Analysis and Design

The interactions between process groups and technology groups are illustrated in Fig. 2. Process groups are intended to apply and test methods and technologies that enhance the performance of the business processes related to their areas of concern. These groups will configure existing technologies such that they can be applied in manufacturing business practice. If it is considered that no appropriate technologies exist, the technology groups will suggest and recommend new technologies which they have been studying, or they will seek to develop new technologies.

On the other hand, the technology groups have the possibility to propose the process innovation to the process groups by applying advanced technologies.

In these research and development activities, we can share the background intellectual property rights and produce foreground information and rights. These interactions will be spiraling and in the final stage of Globeman activities, Globeman demonstration throughout the business life cycle are anticipated.

Because of the wide scope of these groups, it is necessary to form smaller cooperating units to undertake the detailed studies, industrial trials and research. The Work Packages are the operating units which will have clearly defined mandates and specific task areas for each partner to perform.

![Figure 2 Business process & Technology collaboration](image-url)
3.4 Management Structure

Management is very important in such a large consortium. Fig.3 shows Globeman management structure. Board which consists of regional coordinators shall be responsible for political & managerial matters such as ensuring the efficient performance of the consortium, interfacing with ISC, IRS, maintaining the vision and mission of the project. IPR support committee shall be responsible for addressing all legal aspects of the projects. Executive Director shall be responsible for ensuring efficient performance of the project, timely reporting the results to ISC, partners and funding agencies, resolving disputes between partners. Technical Committee shall be responsible to the Executive Director for ensuring the work of the Process Groups and the Technology Groups converge towards the project vision.

![Figure 3](image-url)  
**Figure 3** Management structure of Globeman

3.5 Expected Benefits

Expected Benefits from Globeman activities will be:

- Increased understanding of the key processes in manufacturing, considering an integrated dynamic cooperation between enterprises, and taking account of cultural issues in different regions.
- New management capabilities to operate in a world of global virtual enterprises - i.e. an "enterprise" composed of cooperating companies (not formally incorporated) working together for a specific production task.
- New technologies and new applications in fields such as: modeling, simulation, control, artificial intelligence, team leadership and human organization issues; and the
Part Eight Towards Intelligent Manufacturing Systems

integration of elements from all of these and other disciplines.

- Architectures for more efficient, high quality, production in all domains of manufacture, but particularly in small batch or one-off production. The emphasis will be on a framework of new processes and technologies to provide improved products, more directly focused on customer needs and satisfaction, and with reduced time and cost for product development.

The World Manufacturing Community will benefit, in the long-term, from publication of results and from demonstrations which ensure that Globeman21 influences manufacturing management and business practices throughout the world.

Industrial Partners can gain immediate benefits by becoming part of the developing practices and technologies with shared access to information between leading industrial enterprises and some of the world’s best researchers. Participation in Demonstrators ensures very early benefit and enhances the in-house capabilities for future company developments.

Participating Universities and Research Establishments are given scope for expanding their research activities and undertaking industrial research in association with world class scholars and leading manufacturing companies. Generic results are publishable, while Demonstrators ensure industrial relevance and a global reputation in the real world of manufacturing.

4 PROJECT WORKPLAN

Process Group 1(PG1) - Global Product Life Cycle Management - is aimed at improving the business processes to manage the life cycle of the product. The work of the Group includes description from the conception of the product, all the way to planning and re-use or re-cycling of the product and its materials. Critical to life cycle management is the flow of information through all phases of the product’s life and the seamless transfer of data and knowledge between different companies in all of the phases. Thus, the Work Packages in PG1 include topics such as, defining the life cycle itself, communications within the life cycle and the extended manufacturing enterprise, production and recycling, integration of engineering and production, managing maintenance and renewal of the product/process, and integration of the findings of the whole Process Group into a generic solution that can be applied in manufacturing practice.

Process Group 2(PG2) - Global Manufacturing Management - is directed to the business processes required to manage a globally integrated manufacturing business. It assumes an extended enterprise comprising traditional manufacturing establishments and also including suppliers and customers as part of the involved manufacturing enterprise. Thus suppliers, customers and many other companies are necessarily linked into the total business process. The Work Packages in PG2 include a review of methods to improve the business system in a global operation, development and testing of prototype models for new business processes in repetitive manufacturing and in one-of-a-kind production, consideration of methods for maintaining and using data between diverse groups and a group that will integrate the results of the complete Process Group to achieve a generic demonstrator that can be applied by companies in the consortium and by other companies engaged in global operations.

While the two Process Groups are directed towards different ends - PG1 at the product life cycle and PG2 at the manufacturing enterprise - both assume a global manufacturing interaction, with effective information infrastructure to support the whole operation. There will
be commonality in many of the new technologies required by each process groups. For this reason, the Technology Groups that have been established will link into both PG1 and PG2. There will also be some sharing of tasks and results between PG1 and PG2. So far as possible, the TGs will adapt existing technology to suit the requirements of the Groups. New technology will only be developed when no other suitable technology exists. Demonstrations are a key feature of Globeman. They will exemplify the complete operation of some major element of Globeman. So far as possible, the demonstrations will be real situations and the resulting conclusions will provide an actual improvement in the operation of the company members.

Table 1 Work packages of Process Groups

<table>
<thead>
<tr>
<th>Process Groups</th>
<th>Title</th>
<th>Leader</th>
</tr>
</thead>
<tbody>
<tr>
<td>PG1</td>
<td>Global Project Life Cycle Management</td>
<td>Ahlstrom, Finland</td>
</tr>
<tr>
<td></td>
<td>WP1 Product Life Cycle Process</td>
<td>IPK, Germany</td>
</tr>
<tr>
<td></td>
<td>WP2 Communications Management</td>
<td>TEC, Japan</td>
</tr>
<tr>
<td></td>
<td>WP3 Product Model Management</td>
<td>FhG-IPA, Germany</td>
</tr>
<tr>
<td></td>
<td>WP4 Integration of Engineering and Production</td>
<td>TEC, Japan</td>
</tr>
<tr>
<td></td>
<td>WP5 Operations Support and Maintenance Management</td>
<td>VTT, Finland</td>
</tr>
<tr>
<td></td>
<td>WP6 Management of Renewals</td>
<td>JSPMI, Japan</td>
</tr>
<tr>
<td></td>
<td>WP7 Management by Visual Methods</td>
<td>HUT, Finland</td>
</tr>
<tr>
<td></td>
<td>WP8 Integration of PG1 Results</td>
<td>Ahlstrom/IPA</td>
</tr>
<tr>
<td>PG2</td>
<td>Global Manufacturing Management</td>
<td>BHP, Australia</td>
</tr>
<tr>
<td></td>
<td>WP1 Business System Analysis and (Re)Design</td>
<td>HUT, Finland</td>
</tr>
<tr>
<td></td>
<td>WP2 Management of Repetitive Manufacturing</td>
<td>BHP, Australia</td>
</tr>
<tr>
<td></td>
<td>WP3 Managing One-of-a-Kind Manufacturing</td>
<td>Ahlstrom, Finland</td>
</tr>
<tr>
<td></td>
<td>WP4 Global Data Warehousing</td>
<td>TBA</td>
</tr>
<tr>
<td></td>
<td>WP5 Integration of PG2 Results</td>
<td>BHP, Australia</td>
</tr>
</tbody>
</table>

Table 2 Work packages of Technology Groups

<table>
<thead>
<tr>
<th>Technology Groups</th>
<th>Title</th>
<th>Leader</th>
</tr>
</thead>
<tbody>
<tr>
<td>TG1</td>
<td>Modelling Technologies</td>
<td>Toyota, Japan</td>
</tr>
<tr>
<td></td>
<td>WP1 Modelling Infrastructure</td>
<td>IPK-IWF, Germany</td>
</tr>
<tr>
<td></td>
<td>WP2 Global Product Model</td>
<td>U. Tokyo, Japan</td>
</tr>
<tr>
<td></td>
<td>WP3 Global Business Process Model</td>
<td>RIT, Sweden</td>
</tr>
<tr>
<td></td>
<td>WP4 Tools for Simulation</td>
<td>FhG-IPA, Germany</td>
</tr>
<tr>
<td>TG2</td>
<td>Technologies for Information Access and Control</td>
<td>TEC, Japan</td>
</tr>
<tr>
<td></td>
<td>WP1 Data Communication &amp; Sharing Infrastructure</td>
<td>TEC, Japan</td>
</tr>
<tr>
<td></td>
<td>WP2 Knowledge Sharing Infrastructure</td>
<td>TEC, Japan</td>
</tr>
<tr>
<td></td>
<td>WP3 Tool Integration</td>
<td>Yokogawa, Japan</td>
</tr>
<tr>
<td>TG3</td>
<td>Technologies for Scheduling and Coordination</td>
<td>CMU, USA</td>
</tr>
<tr>
<td></td>
<td>WP1 Constraint-Based Scheduling</td>
<td>CMU, USA</td>
</tr>
<tr>
<td></td>
<td>WP2 Coordination Methods</td>
<td>CMU, USA</td>
</tr>
<tr>
<td></td>
<td>WP3 Generic Agent Shell</td>
<td>U. Toronto, Canada</td>
</tr>
<tr>
<td></td>
<td>WP4 Workflow Management</td>
<td>U. Toronto, Canada</td>
</tr>
<tr>
<td>TG4</td>
<td>Technologies for Business Process Analysis and Design</td>
<td>CSIRO, Australia</td>
</tr>
<tr>
<td></td>
<td>WP1 Methodology for Business Process Analysis and Design</td>
<td>Griffith U., Australia</td>
</tr>
<tr>
<td></td>
<td>WP2 Business Process Analysis and Manufacturing Strategy Formulation</td>
<td>CSIRO, Australia</td>
</tr>
<tr>
<td></td>
<td>WP3 Alternative Approaches for Business Process Analysis and Design</td>
<td>TBA</td>
</tr>
<tr>
<td></td>
<td>WP4 Visualising Business Process</td>
<td>TBA</td>
</tr>
</tbody>
</table>
5 RESULTS AND ACHIEVEMENTS

The results of the project fall within the following items:
1. Partner requirements
2. Industrial scenarios
3. Demonstrators
4. New products
5. New processes
6. New competence and skills
7. New IT systems and tools
8. New methodologies
9. International collaboration

6 CONSORTIUM PARTNERS

Currently 46 partners signed Consortium Corporation Agreement from all regions. The people resources applied to Globeman over the next 3 years will be order of 4500 person-months. This is actually 40 million US$ project for 3 years.

International Coordinating Partner:
Newport News Shipbuilding

Regional Consortium Partners:

Japan Region
Coordination Partner: Toyo Engineering Corporation
Industrial Partners: Toyo Engineering Corporation, Omron Corporation, YokogawaElectric Corporation, Daikin Industries Ltd, Toyota Motor Corporation, Mazda Motor
Enterprise integration for global manufacturing

Corporation, Ricoh Company Ltd, Takenaka Corporation, Mitsui Engineering & Shipbuilding C.Ltd, Toyoada Machine Works

Academic Partners: Electrotechnical Laboratory, Japan Society for the Promotion of Machine Industry, Institute of Industrial Science, University of Tokyo, Nagoya University,

European Union(EU) /EFTA Region
Coordination Partner: A. Ahlstrom Corporation, BICC
Industrial Partners: A. Ahlstrom Corporation, BICC, Logistics Support Consultants, Intracom SA, Odense, Partek, YIT, NCR Norge AS
Academic Partners: Bremen Institute of Industrial Technology, FhG-IPA, IWF, VTT, IIA-Research Centre Helsinki University of Technology, Royal Institute of Technology, Technical University of Denmark, SINTEF, Technical University of Hamburg-Harburg.

Australia Region
Coordination Partner: BHP Pty Co.
Industrial Partners: BHP Pty Co., Farley Cutting Systems Pty Ltd, Moldflow Pty Ltd.
Academic Partners: Commonwealth Scientific and Industrial Research Organization, Griffith University

CANADA Region
Coordination Partner: University of Toronto
Industrial Partners: Northern Underwater Systems, Axion Spatial Imaging
Academic Partners: Simon Fraser University, University of Toronto, University of Alberta

USA Region
Coordination Partner: Newport News Shipbuilding
Industrial Partners: Newport News Shipbuilding, Deneb Robotics
Academic Partners: Carnegie-Mellon University, University of Virginia

6 CONCLUSIONS

Globeman has so far successfully integrated industrial and academic people from 4 continents. This is achieved by developing common IT infrastructures, objectives, methodologies and approaches.

The efficiency of this size of project has surpassed our expectations. Cultural and other differences among partners do not seem to prohibit creative collaboration as long as we share common vision, concepts, infrastructures and pattern of behaviors.

The importance of global cooperation and collaboration can not be overemphasized. Most governments contribute to programs to give aid to the underdeveloped countries. Globeman is developing technologies that will open opportunities to improve the way this aid can be provided. Sharing knowledge, competence and skills and transferring results and values can be achieved without cultural and human interference and with simultaneous interaction.

The Globeman partners are satisfied with the project and have great expectations to the results being continuously developed and deployed.
Next Generation Manufacturing Systems in the IMS Program

T. Kurihara
Fuji Electric Co., Ltd
Tokyo 100, Japan
Tel: +81 33536 8825 Fax: +81 33201 0750
kurihara@fujielectric.co.jp

J. Jordan
Consortium for Advanced Manufacturing - International
3301 Airport Freeway, Suite 324
Bedford, TX 76021 USA
Tel: +1 817 860 1654 Fax: +1 817 275 6450
jajordan@ix.netcom.com

P. Bunce
Consortium for Advanced Manufacturing - International
20 Market Street
Poole, Dorset, BH15 1NF, UK
Tel: +44 1202 670717 Fax: +44 1202 680698
peter@cam-i.demon.co.uk

Abstract
The International Intelligent Manufacturing Systems (IMS) Program is an important program of cooperative research projects, one of which is the Next Generation Manufacturing Systems (NGMS) project. NGMS seeks to develop the technologies and methodologies needed for the manufacturing systems that will support the next generation of manufacturing enterprise. NGMS is integrating thinking on advanced manufacturing systems from Europe (the fractal factory), from Japan (autonomous and distributed manufacturing systems, biological manufacturing systems) and the United States (agile manufacturing). We give an overview of requirements for NGMS and a summary of the applied research the project is undertaking.
Keywords

1 THE INTELLIGENT MANUFACTURING SYSTEMS (IMS) PROGRAM

The IMS Program was conceived in Japan in 1989 as an international, industry driven program of collaborative research and development (R&D). After negotiations, the Japanese proposal was accepted on a provisional basis and a set of test cases begun in 1993. After further negotiations, Australia, Canada, Japan, Switzerland and the United States agreed to Terms of Reference (ToR) for a full scale, 10 year program beginning in 1995. (As of June 1996, the European Union is expected to ratify the ToR). In parallel, the Japanese IMS Promotion Centre established a Domestic Japan IMS Program. In September 1995, the IMS Steering Committee endorsed the first full scale projects. The IMS Program is comprehensive, with major technical themes that span the needs of manufacturing enterprises for the early Twenty First Century:

- Total Product Life Cycle Issues, including future general models of manufacturing systems.
- Process Issues, including process technology innovation, more flexible and more autonomous processing modules, and better interaction and harmony among various components and functions.
- Strategy/Planning/Design Tools, including methods and tools for business process re-engineering, to support the analysis and development of manufacturing strategies, and to support planning in an extended enterprise or virtual enterprise environment.
- Human/Organization/Social Issues, including improved capability of manufacturing workforce/education, training, autonomous offshore plants, and corporate technical memory.
- Virtual/Extended Enterprise Issues.

2 NEXT GENERATION MANUFACTURING ENTERPRISES (NGMES)

Next Generation Manufacturing Systems (NGMS) will support Next Generation Manufacturing Enterprises (NGMEs). NGMEs have been characterized (Jordan, 1994a) as dynamically combining customers, multiple design and production entities, and suppliers, into organizations that will form to meet a customer need, fulfill the need, and then dissolve. NGMEs are expected be the dominant form of manufacturing enterprise in a time of unpredictable competitive challenges and a rapidly, chaotically, changing global business environment.

The important elements of Next Generation Manufacturing Enterprises are:

- NGMEs will be customer-driven. Customers will be deeply integrated into all aspects of the product cycle.
- Suppliers will be integrated into the product cycle. Sub system suppliers, especially, will become peers.
- As illustrated by much of the thought about NGMEs, e.g. the thinking underlying the Japanese Autonomous and Distributed Manufacturing Systems and Biological
Manufacturing Systems efforts (IMS93-II-1 Group, 1994, the Fraunhofer Society's Fractal Company (Warnecke, H.-J., 1993), and the U.S. Agile Manufacturing activities (Nagel, R., et.al. 1991), and the view of the factory floor; rigid, static, hierarchical, manufacturing enterprises will be replaced by virtual enterprises exhibiting great adaptability to rapid change and able to produce small lots with high quality and at low costs.

- NGMEs will be made up of simple, distributed, autonomous but cooperating, work units, that will work in flattened, network like, organizations.
- The global economy and the technologies for telecollaboration will both require and enable work units to be distributed globally.

3 INTRODUCTION TO NGMS

NGMS will support the product life cycle within NGMEs. That is, NGMS will integrate a dynamically changing collection of self-organizing, autonomous but cooperating, distributed work units executing the processes that relate specifically to products and their development, production, distribution, maintenance, field enhancement, and disposal. These processes will have to be integrated and supported at the Enterprise Level, Factory Level, and the Factory Floor. (Additional systems, integrated at a high level with NGMS, will support the other processes of an NGME.)

Key Characteristics of NGMS

- Support for virtual enterprise
- Customer focused and business centred
- Reconfigurable, adaptable, and flexible in response to customers
- Global support for design and production
- Information and knowledge based, human intelligence oriented
- Modular to support distribution and autonomy, but cooperative to achieve enterprise goals
- Environmentally aware

Key characteristics of NGMS were developed at an NGMS Program Definition Workshop (Jordan, 1994b) conducted in February 1994 for twenty large manufacturing companies from Europe, Japan, and the United States. The Workshop also identified a longer list of needs that the NGMS should meet.

4 NGMS ARCHITECTURE

Four concepts: Agile Manufacturing, the Fractal Company, Bionic Manufacturing Systems (BMS) and Autonomous and Distributed Manufacturing Systems (ADMS) provide the basis for the NGMS architecture. Each of the concepts being developed in different parts of the world, contributes to meeting the requirements of highly adaptive NGMS able to support competitive, agile, manufacturing enterprises. Each assumes that NGMEs will be organized into distributed work units with a high degree of autonomy and intelligent behaviour. The concepts deal with different aspects of manufacturing systems; the R&D will reach fruition at different times, but the combination of these four views is a powerful, representation of advanced manufacturing systems.
Needs NGMS Must Meet:

- Be a Global System.
- Be a Portable System.
- Be an Integrated System.
- Be Supportive of Integrated Product/ Process Design (IPPD).
- Support and Use Human Intelligence.
- Support Intelligent Machines.
- Provide "Just-in-Time" Information.
- Be Agile.
- Be Model and Simulation Based for Virtual Manufacturing.
- Be Environmentally Oriented.
- Include Process Control

NGMEs will have fewer levels of hierarchy and will have information systems capable of conveying floor level information throughout the enterprise. Time constraints will make it important for the enterprise to have an accurate understanding of the status of the floor level operations in order to make timely decisions affecting enterprise level activities. Because of this, NGMS will be more tightly integrated across the floor, factory, and enterprise levels. It will be difficult to decompose NGMS in the traditional hierarchies or levels and so it is important to take the best ideas at all levels and bring them together in a unified view of NGMS.

Agility provides a philosophical basis for NGMS. Agility speaks to the capabilities of an enterprise to reconfigure itself quickly in response to sudden changes, but in ways that are timely, cost-effective, of a broad scope, and robust. Agility theory seeks to provide metrics for business processes, for physical operations, and for human resources to respond to rapid and unpredictable change. The emphasis on agility implies that time must be treated very carefully in NGMS models and simulations, and that NGMS must include instrumentation and analysis tools for work unit, factory, and enterprise level measures.

The Fractal Company describes an organization, made up of self similar, self organizing, autonomous work units (fractals). A strength of the Fractal Company concepts is the guidance they give to business process re-engineering, to the propagation of goals, and to the human element in NGMEs. Although work units will have wide latitude about how they accomplish their tasks in the virtual manufacturing enterprise, they will have to align their goals with those of the enterprise. Fractal Company R&D is building a manual methodology for goal setting and propagation (termed navigation) in enterprises organized into empowered work teams. This methodology appears also to be applicable in more loosely coupled enterprises.

Autonomous Distributed Manufacturing System (ADMS), suggested in Japan, aims to realize the autonomous distribution of modules of manufacturing system, by giving intelligence to each of the modules. Here the manufacturing system is composed of module units, which are functioning autonomously and cooperatively, and are integrated into a virtual production system.

Biological Manufacturing Systems (BMS), which is the further advanced concept of ADMS, have the functions imitating those of biological organisms, such as self organization, self
recovery, self growth and evolution, and will provide the methodology covering all the levels. Here we intend to realize the manufacturing system which can quickly respond to needs and is harmonious to natural environment, by systematizing the information of a product throughout its whole life cycle, which consists of planning, design, production, consumption and disposal.

In this research we will try to develop decision support systems and architectures which provide common frameworks for manufacturing systems in the course of transition. These will change from the current concentrated manufacturing system to an autonomous distributed manufacturing system, and further on to a biological manufacturing system.

![Diagram](image)

**Figure 1**

5 THE NGMS IMS PROJECT:
DESCRIPTION, MODELLING AND SIMULATION OF NGMS

In late 1993 an international partnership of leading manufacturing companies, supported by a strong group of research universities, came together to set an R&D agenda whose results will be the technologies, methodologies, and sub systems needed to transform today's manufacturing systems into the ones that will best support NGMEs.

Building on an excellent base of work in Europe, Japan, and the United States, the partnership developed a comprehensive R&D agenda that spans the product life cycle and all key issues in manufacturing. Working through the agenda will take a decade.

The first set of tasks on the agenda form the basis for a proposal made by the consortium to the IMS Program, a proposal called Description, Modelling, and Simulation of Next Generation Manufacturing Systems (NGMS): Merging the Agile, Autonomous and Distributed, Biological, and Fractal Manufacturing Systems Concepts. The proposed project was endorsed by the International IMS Steering Committee in September 1995 and work begun at the NGMS International Conference, held in Irvine, California, USA, in February 1996.
The goals of the NGMS IMS Project are to:

- Develop a unifying description of NGMS, an NGMS Specification that captures the results of the individual R&D activities, and a framework for ensuring the integrability of the results into cost-effective NGMS.
- Develop on-line facilities for tracking and presenting advanced technologies and processes and advanced materials that will be used in and by NGMS, gauging their readiness for application.
- Develop an integratable set of models and simulations merging a bottoms-up view of the factory floor as it will be found in NGMEs with a top-down view of the globally distributed virtual enterprises.

The unique strength of the NGMS IMS effort is its systems approach. Starting with the NGME vision, the effort has adopted a needs-based understanding of the characteristics of future manufacturing systems and has defined an R&D agenda to develop the best ideas on advanced manufacturing systems and integrate them into NGMS. The key issues of NGMS have to do as much with the integrability of manufacturing technologies and processes.

6 WORK PACKAGES AND TASKS

The Project objectives will be accomplished in a set of tasks clustered in three Work Packages.

6.1 Work package 1

This Work package will provide the framework for the NGMS IMS effort.

Task 1.1. Description of NGMS will provide a standard description of NGMS, with key words defined and key concepts described, using the four central concepts, augmented with additional ideas on advanced manufacturing systems. Where different words are used to describe similar concepts, a mapping will provide a shared understanding of the vocabularies used to articulate the concepts.

Task 1.2. Specification of NGMS will maintain the NGMS Specification as a timely and complete documentation of the vision and functions of NGMS and as the definitive statement of the context in which the NGMS IMS Program's Work Packages and tasks will be pursued. The updated Specification will become a progressively more detailed description of NGMS as the results of a succession of Work Packages performed by the NGMS IMS effort are integrated into it.

Task 1.3. NGMS Systems Integration has two sub tasks. In the first, cross-Regional task team is developing and maintaining an NGMS systems integration framework, considering both horizontal integration (e.g. the things that relate to the floor level) and vertical, integrating functions at the floor, factory, and enterprise levels. The task team will identify inconsistencies and ambiguities among the Work Packages; where appropriate, it will recommend interface specifications for ensuring NGMS integrability. In the second sub task, the cross Regional task team will identify requirements for one or more systems integration testbeds where the
integrability of NGMS IMS Program work products can be evaluated.

6.2 Advanced Knowledge in Manufacturing Systems

We will speed the application of advanced technologies, methodologies, systems, and materials by developing publicly accessible knowledge bases usable by process engineers. The intent is to package knowledge developed by the partners in the NGMS effort, by other IMS projects, and from other sources into an on-line system, called NGMnet. The knowledge bases will include a description of the innovative technology, methodology, system, or material, implementation and experience information, and an assessment of the risk of adoption. A second sub-task will develop an interactive, on-line, Handbook of Standard Fixes. New technologies often are buggy, but be made reliable and useful when used in restricted ways or with the application of a small patch. The discovery of the fixes to the bugs can be a time-consuming process that is repeated as companies attempt to use the new technology. The Handbook will provide a vehicle for process engineers to record and propagate the fixes they discover and for other process engineers to access fixes as (or before) they encounter bugs.

6.3 Work package 3

There are four tasks involving modelling NGMS from four different perspectives. These four tasks will be conducted primarily as Regional tasks; a cross-Regional task will ensure that the tools used (e.g., object-oriented modelling tools) are consistent and that resultant models present a consistent representation of NGMS.

Task 3.1. Modelling and Simulation of Agile Manufacturing Systems is derived from on-going work in Autonomous and Distributed Manufacturing Systems (ADMS) being conducted under the Japanese Domestic IMS Program Task II-1. In this task we are developing

- Position of ADMS in NGMS

ADMS aims to fulfill the characteristics that are required in NGMS, such as flexibility, quick response, adaptability, globally, and concurrency. The system configuration of ADMS is autonomous and distributed, and its aim is cooperation and harmony. Viewing ADMS from NGMS as a whole, it focuses mainly on the production phase among the life cycle, which includes development, design, production, physical distribution and post-sales, as shown in Fig 4. The subject of the research is modelling and operation in the phase.
• Modelling of ADMS

Modelling provides the basis for the realization of ADMS, and it corresponds to the architecture of information processing. In the research on ADMS, we develop three kinds of modelling tools. They are STN(Scene Transition Net), Agent Net, and Job Model, as shown in Figure 5. Their common bases are the object-oriented technology, the discrete system theory, and the dynamic system theory.

STN is composed as a hybrid system that is able to integrate and deal with both continuous and discrete events. It aims to take in and integrate models that are based on even more different aspects, and to perform a wide range simulation of manufacturing system.

Agent Net aims to be applied in real-time control and scheduling, by merging Petri Net and object-oriented technology, and combining functions of cooperation, learning and self-organization.

Job Model aims to be applied in intelligent communication, which is to support computer-aided manufacturing in autonomous and distributed way. It attempts to take human factors into traditional product models and factory models.
Operation of ADMS
Operation refers to the decision support on manufacturing system, which is realized on the basis of modelling technology, and involves actions such as communication, control, and scheduling.

(a) Autonomous Distributed Scheduling
Development of a scheduling system which has the function to pursue self optimization in each individual process, and to simultaneously cooperate with other processes and aim for total harmonization.

(b) Autonomous Distributed Control
Development of control technology for autonomous distributed manufacturing system, which has robustness to troubles, flexibility, and easiness for construction, by using Agent Net.

(c) Intelligent Communication
Development of intelligent ways of communication to realise HIM (Human Integrated Manufacturing), which is an advanced form of CIM that harmonizes machines and humans, by using Job Model.

Task 3.2. Next Generation Enterprise Modelling, Simulation, and Operations will develop new forms of enterprise models that will provide assistance in the formation, transition and management of NGMEs. The task will focus on the relationships and communications between individual autonomous work units when they are either participating in a single enterprise or participating simultaneously in several virtual enterprises. This task will focus on identifying the right partners and the right interfaces among partners in an NGME.

Task 3.3. Modelling for Biological Manufacturing Systems will establish basic models for Biological Manufacturing System (BMS). It is well-known that for NGMS, a manufacturing
system with autonomous distributed function is required. However, it is not always the right way to construct it as an extension of traditional method. The methodology of system construction is an important point, for it is inevitable for NGMS to be in harmony with society and nature.

Figure 4

BMS is 'a manufacturing system learning from and living with biological life'. By introducing into artifacts the excellent ability of biological life, such as self recognition, self growth, self recovery, evolution and adaptation, BMS conceives interactions between human beings and artifacts, and furthermore, tries to locate artificial system in the macro ecosystem, covering the whole life cycle of a product, i.e. planning, design, production, operation, maintenance, recycling and disposal.

Figure 6 shows the outline of the stage configuration and the information flow of BMS. Figure 7 shows the entire view of the research on BMS. Considering the life cycle, its topics can be listed up as follows:

(1) BMS Core System
Research mainly based on the way to construct a Biological Product Model (BPM), which is a key to BMS, in an attempt to obtain a basic mechanism by which biological characteristics are applied.

(2) DNA-Oriented Design System
Research on evaluating-type design by using biological product model.

(3) Biological Information Processing Function
Research on information processing function that biological facilities are supposed to have, focusing mainly on production stage.
(4) Product Life Cycle Feedback
Research on heredity and evolution of a product.

(5) Harmonization with Macro Ecosystem
Research on total life cycle including disposal and recycling, by using simulation, etc.
Although the above research topics are closely related with one another, we intend to concentrate mainly on the topics (1), (2), and (3) for the time being.

![Diagram of product life cycle and harmonization]

Figure 5

Task 3.4. Modelling for Virtual Enterprise will develop models to help in the formation and management of virtual enterprises, considering both the enterprise as an entity and individual work units that may be participating in several virtual enterprises. The models will illuminate the decision points in the enterprise life cycle; e.g. to illuminate the decision to combine to offer a product at a competitive target price and to assist individual autonomous work units in their decisions to commit to participation in multiple enterprises.

Task 3.5. Modelling Tools and Model Integration will establish the mechanism to integrate the four modelling tasks. There are two major integration sub-tasks: tools and models. Each assumes that the modelling tools will be based on the object-oriented programming paradigm and each has object oriented tools under development. An objective of this task is to ensure that the tools are compatible, that their semantics and interfaces are consistent. A set of NGMS IMS Program standards, that will inform the establishment of international standards and conform to them once established, will be developed by a cross Regional task team. Each of the modelling tasks assumes that work units will conduct negotiations as they cooperatively reach decisions relating to the enterprise’s goals and their individual roles in meeting those goals. A cross Regional task team will find the commonalties among the algorithms and methodologies used in the other tasks to find optimizations and to help establish standards.
As the four concepts of Agile Manufacturing, Autonomous and Distributed Manufacturing System, Biological Manufacturing System, and Fractal Company mature, each will contribute toward a unified view of NGMS. Our assessment of the way these concepts will mature over time is shown in Figure 8. At any given time, the next generations of manufacturing systems will be a combination of the most useful ideas coming from the four concepts. The combination will change over the lifetime of the NGMS IMS Program.

**Figure 6**

7 OPERATION AND MANAGEMENT STRUCTURE FOR THE INTERNATIONAL COOPERATIVE RESEARCH

The NGMS IMS Program is organized as follows:

**Figure 7**

Coordinating Partner  I. International: CAM-I NGMS Program Office
A. Australian Group: Expected to join in the near future
E. European Group: CAM-I European Office
J. Japanese Group: Fuji Electric Co., Ltd.
U. The Group in the USA: CAM-I NGMS Program Office
The administrative management of this IMS Project is the responsibility of the NGMS IMS Program Office, as international coordinator, and the Regional Coordinating Partners. The Program Office is responsible for coordinating the inter Group aspects of the Program and will be the Program's liaison with the International IMS Steering Committee and with the International IMS Secretariat.

8 SUMMARY

The first results from the NGMS R&D program will be available in 1997; others will flow in the next two years. Taken together, the results will lead to a transformation of the NGMS partners' manufacturing systems into those that can support fast moving global enterprises in rapidly changing and very competitive markets.

9 ACKNOWLEDGMENTS

The requirements of Next Generation Manufacturing Systems are a synthesis of contributions made in NGMS Program workshops and conferences by the NGMS project partners and other thought leaders from many manufacturing companies and universities.
Holonic Manufacturing Systems

M. Höpf, C.F. Schaeffer
Fraunhofer Institute
Manufacturing Engineering and Automation
Nobelstrasse 12  70569 Stuttgart, Germany
Fax: +49(0)711 970 -1008
Phone: +49(0)711 970 - 1127
e-mail: hoepf@ipa.fhg.de

Abstract
This contribution introduces in the background of holonic systems and holarchies as a hierarchical model for biological and social structures in nature. The characteristics of these hierarchical architectures and its elements will be explained. It will be shown how this approach can be used as a potential solution for present and future challenges in manufacturing control systems. The results of an internationally performed feasibility study heading for the joint development of this technology will be described. A report on the current status of the international research activities will cover the technical and organisational aspects of the project.

Keywords
Holon, manufacturing, IMS, control system, architecture.
1 THE BACKGROUND OF HOLONIC ARCHITECTURES

The History of Holarchies

The expressions „holon“ and „holarchy“ first appeared in 1967 in Arthur Koestlers book „The ghost in the machine“. Koestler was a bulgarian born journalist and spend most of his life with biological and social studies to explain the human fate. „The ghost in the machine“ is the last book of a trilogy where he analyses biological and social evolution to proof that the historical behavioristic approach with its mechanistic world-view is not an appropriate model for the organisation of living structures. It was very important for him to show that human beings are not a kind of biological robots whose behaviour is predetermined by their role in fixed hierarchies. Instead of characterizing a system as a whole and the elements as dependent parts of it, Koestler found out that the functional properties of social phenomena as well as biological structures require that the elements have tendency to integrate themselves in a structure but at the same time try to preserve their autonomy as self-contained independent units.

Arthur Koestler's motivation for this kind of research was his own fate. Born in 1905 he past most of his life in Europe, Palestine and Russia as a journalist and writer of novels and essays. His engagement with communism and the fact that he was a jew was the reason that he was put under pressure from various authorities and he even had to pass several years in the jail in Spain. There he started to research the structures of biological life by analyzing the interaction of organs in animals and the human body. He compared the structures he found with social systems and developed the open hierarchical systems (OHS) as a general organisation principle of life.

Figure 1 Architecture of natural systems.
The System „life“

The model Koestler developed for the system „life“ is based on the fact that hierarchies are an essential element in living structures but these hierarchies are not fixed. Living organisms obviously can change their structures and are dissectable. This enables the systems to use in-build deficiencies and defects to optimise their function according to present objectives as for example survival in dangerous situations. Hierarchies in nature are therefore temporary. This is more obvious in social systems where the ability of self-adapting hierarchies is a basic principle of democracy. Fixed subordination as functional element does not exist in nature (biological and social structures). The organisation of living systems is characterized by a constantly changing network where the elements play the role of exchangeable nodes and the links are represented as interactions like communication, negotiation and even aggressions. These interactions can be based on external or internal events and always result in a change in the temporary hierarchical structure and subordination scheme.

Evolution and social development are the logical consequence of this self-organising systems, where even the top of hierarchies are not fixed. Systems which do not have these properties are not stable in nature. Evolution and survival depend on a constant transition in structures. This requires, that all elements of the system are not designed to take over a predetermined role in a hierarchy but are autonomous and self-similar. Living systems cannot be divided in the whole and parts of it or in components and the system. Elements that have this autonomy are called „Holons“ and their behaviour is described by Koestler in a set of rules. The architecture of systems which are composed of Holons is called „Holarchy“. Living systems are either unstable or holarchies.

Holonic Systems Characteristics

One of the most important characteristics of Holons is the Janus-Effect. While the Holons display autonomy they tend also to integrate themselves into a hierarchy. If Holons are given the possibility of interaction and rules for this interaction they will automatically form a system and attract further holons for integration. Thus holonic systems are self-developing and have the inherent tendency to grow. Since the holons are autonomous the systems are dissectable. Parts of the systems or individual holons may be separated or organised in different holarchies. The holons try to maintain a dynamic equilibrium between their integrative and autonomous tendencies. If this equilibrium is disturbed, the system tries to reorganize itself. The goal of this reorganisation is the adoption of the network to changed conditions and the optimized function of the system. This process is initiated by the holons tendency to cooperate.
2 HOLONIC SYSTEMS IN MANUFACTURING TECHNOLOGY

New Challenges in Production

Industrial manufacturing is at the beginning of a technical revolution. Fast innovation cycles, complex products and decreasing product life cycles are the new challenges in competitive global manufacturing. Manufacturing is changing more and more to a self-contained business which has to be effective and less expensive than today. The fast development of new products and processes becomes a problem for large series lines and fixed production installations. The costs for installation changes and reconfiguration in a line reach one third of overall costs. The reliability of hierarchical organized manufacturing systems becomes uncontrollable if the systems get too complex.

New enterprise organisation schemes like the fractal factory are already in course of implementation as a reaction on these environmental changes.

On the technical level of production control hierarchical structures are still standard. Decentralized architectures are currently under development but still their integration effort is too high to be competitive. Moreover any change in the system is has to be regarded as a disturbance or even as a breakdown, but today the number of changes caused by the environment requires a constant transition of a line during operation.

The current level of automation can only be maintained if these problems are solved and the control systems of manufacturing systems support constant and on-line change and
configuration with reduced manual effort. The objective is to lower the costs for electrical installations and dedicated software developments significantly.

Holonic Manufacturing Systems as the result of configuring the controls of individual manufacturing equipment as holons is the key technology to reach this objectives.

**Industrial Benefits**

- reduce IT investments in manufacturing lines
- reuse control equipment and software
- maximize fault tolerance in complex manufacturing systems
- reduce the cost of changes

*Figure 3 Industrial benefits of industrial controls.*

**CIM and Holonic Systems**

The development of computer integrated manufacturing has shown that automation has a similar potential for effectiveness in production as the use of computers in business administration. CIM has developed from single machine control and programmable logic controllers to overall production control systems based on information networks. New developments are aiming at open systems and standardized communication channels with the possibility to transfer and process all kind of information on the shop floor. The controls of the process equipment have access to all informations in real time and are able to communicate not only with a central system but with each other. At the same time the capabilities for decentralized complex data processing improve dramatically and the individual control system can be configured as autonomous systems at a reasonable price. What lacks is a common concept to use this functionalities to overcome the problems encountered. In that view holonic manufacturing systems is an implementation strategy for the further developments of CIM. HMS can be considered as a software technology on the top level of the communication
infrastructure in networked manufacturing machines enabling the use of large information resources for decentralized decision making.

The use of extended information technology in machine controls in conjunction with information about the current production and business strategies results in a self configuration behaviour avoiding extensive efforts for programming and manual definitions of actions and reactions. This can be compared with the self configuration capabilities of computer ad-ons replacing the manual configuration tasks of the respective electronic boards. It is estimated that this functionality will save up to 30% of the installation costs for a new manufacturing line.

By adding on-line information about the current status of individual machines and the production line to this network, the reliability and availability of complex manufacturing systems is enhanced. In case of a defect or even unforeseen disturbances in normal operation of a line, the analyzing and negotiation capability of the controls is used to investigate auxiliary options in the process and to configure optimal alternatives to maintain the production process.

Due to this qualities holonic control systems are the basis for the effective use of advanced computer integrated manufacturing and industrial communication technologies.

![Figure 4 Implementation of holonic control systems.](image)

**Development aspects of Holonic Manufacturing Systems**

Holonic control systems as the basic technology for holonic manufacturing systems will be based on a number of innovative technical approaches which are currently under development.
One of the most important technical components for a holonic systems is the communication architecture. The field bus systems today available have to be evaluated for their qualification in co-operative networks. The standardisation of the functional architecture is already an ongoing activity.

Holonic Manufacturing systems are the logical consequence of these emerging technologies. With the availability of the communication networks the question what to do with it arises. Just transferring programs and commands or supervising machinery and systems would not justify the expensive equipment and software. Even the co-ordination of complex schedules does not require extensive communication. In this light, HMS adds a new type of functionality to the individual manufacturing component by using the communication for autonomous interaction. This concept is based on the scientific theory of agent systems. This theory provides the algorithms for negotiation and decision making in artificial intelligence. HMS adds the rules and common contracts to such a system in the context of a dedicated manufacturing system.

3 THE INTERNATIONAL HMS PROJECT

In 1993 a feasibility study was commenced to evaluate the possibilities of international co-operation for the development of HMS. This study was one of six test-cases launched under the international IMS programme and was carried out by a consortium of 31 partners from North-America, European Union, Australia and Japan. The study was focused on the investigation of the industrial potential of HMS and the outline of a work programme for a joint development project. Starting with the research on user requirements and forecasts for future trends in manufacturing the consortium also developed three testbeds to evaluate the characteristics of HMS in real applications. The results of the feasibility study encouraged the 15 industrial participants to agree on a joint project to develop, produce and market a new type of industrial controllers in a world-wide co-operation.

### Project Overview

**Generic Workpackages:**
- Architecture & Engineering
- Systems Operation

**Application Workpackages:**
- Resource Management
- Manufacturing Units
- Fixturing
- Handling
- Mobile Equipment

*Figure 5 The HMS Project.*
In the following interim phase the consortium was restructured and a contract framework was developed. On the basis of the Terms of Reference as an international background a consortium co-operation agreement was negotiated. This agreement basically regulates the creation and use of intellectual property within the consortium. A set of workpackage agreements include the workprogrammes and the contributions of the consortium partners.

The workprogramme comprises two generic workpackages and five application workpackages. The generic workpackages will develop the design technology and the basic function in close co-operation with current standardisation activities. The application workpackages are will implement these concepts in primary industrial areas identified in the feasibility study. Extensive technology interchange mechanisms will ensure that application experiences are evaluated in parallel by the generic workpackages and the basic design and architecture will be compatible in the applications.

The contracts and the workprogramme have been accepted by the international steering committee of the IMS Programme in late 1995. The consortium is now preparing the start of the development project.

4 BIOGRAPHIES

Christoph F. Schaeffer

Michael Höpf was born in 1955 in Saarbrücken, Germany. In 1983 he concluded his study in technical and economical Cybernetics in Stuttgart as Diplom-Ingenieur. In the same year he started working as a junior scientist at the Fraunhofer Institute for Manufacturing Technology and Automation, IPA. His working area was robotics application in machining. In 1988 he became group leader in the department of sensor technology and Robot Systems at the IPA. There he was responsible for various industrial contract developments in the area of sensor systems and industrial software. He was also active in public founded consultancy projects to support SMEs in implementing new manufacturing technologies. First experiences with the initiation and the management of EU-founded projects where gained in Brite-Euram and ESPRIT Projects where the institute took over the role as a co-ordinator.

Since 1993 he participates in the HMS Project where he became official delegate in the management committee in 1994.
Abstract
GNOSIS is one of the founding research consortia in the international Intelligent Manufacturing Systems research program. GNOSIS aims to establish the framework for a new manufacturing paradigm through the utilization of knowledge-intensive strategies covering all stages of product life cycle, in order to realize new forms of highly competitive manufactured products and manufacturing processes which are environment-conscious, society-conscious and human-oriented. The project objectives, current situation and its background are described in this paper.

Keywords
Manufacturing, IMS, GNOSIS, New paradigm, Knowledge systematization, Soft artifacts, Virtual manufacturing, Enabling technology, Knowledge highway, Knowledge capitalization, Modeling, Life cycle, Knowledge intensive engineering

1 INTRODUCTION
The role of manufacturing has recently been brought into question due to the undesirable effects it causes -- environmental problems such as natural resource depletion and excess waste generation, and international trade friction as evidenced by the emergence of trade
blocks and the necessity for complex trade agreements. Reactive and uncoordinated remedies are being applied to these problems, achieving only short-term, partial success. Without a coordinated, radical initiative, the manufacturing sector will be faced with either self-imposed or externally-imposed restrictions in the coming years. The work proposed here is an initial but significant step in this initiative.

Based on its experience in the international IMS Feasibility Study, the GNOSIS Consortium has established a research program consisting of 5 work packages focusing on research themes considered essential for the achievement of the above aims. Knowledge systematization forms the basis on which the work will be carried out, supporting enabling technologies and research into the development of soft artifacts (socially desirable, reconfigurable and reusable systems, components and products) and virtual manufacturing through distributed, networked enterprises. These new products and production systems will be designed using knowledge intensive engineering in order to realize the long term aims of the work, namely, the post mass production paradigm.

2 BACKGROUND AND STATE OF THE ART

Although advances brought about by the present information society have improved many aspects of human life, they have not addressed the fundamental problems associated with mass production -- excessive use of natural resources and pollution. Companies are now facing new challenges on top of conventional concerns such as quality and cost. They have to respond quickly not only to changing market demands, but also to environmental pressures and international trade issues. To solve the these problems, the influence of manufactured products on their natural environment and on the global economy has to be investigated. This requires an expansion of the conventional limits of the manufacturing domain to include the entire product life cycle. These issues have not been addressed by conventional information technologies, which tend to focus on data and information and do not have a systematized approach to the treatment of knowledge. In the manufacturing domain this has resulted in localized but not global consistency, and advances in specific areas of product development, but not comprehensive thinking about life cycle issues.

In order to change the current production paradigm, we have to reformulate manufacturing concepts, through comprehensive use of knowledge. Knowledge, of itself, will not solve the problems. Rather, effectively systematized knowledge applied to achieve new, innovative technologies is needed to realize the shift in paradigm. With the aim of design for complete product life cycle, methodologies for realizing new soft, reconfigurable artifacts based on reusable components are being studied in GNOSIS. New production facilities and concepts, which are necessary to achieve these soft artifacts, are also being investigated, as the core of research into the virtual factory.

GNOSIS is taking a unique approach toward overcoming current problems in the manufacturing world which have been and are being tackled by several other projects both within and without the IMS Program. Firstly, GNOSIS is distinguished by its emphasis on the need for a paradigm shift, while other programs up to now have tended towards extending the current paradigm and merely trying to improve it. Secondly, other manufacturing research has generally focused on specific issues or domains, and has not tackled the underlying issues. As noted in the reports of the Test Case, GNOSIS has and will continue to tackle the
problems in a holistic way, not separating design from manufacturing, and manufacturing from environment.

3 PROJECT OBJECTIVES, RESULTS AND OVERVIEW

3.1 Objectives and Industrial Relevance

The GNOSIS Project is a result-oriented, proactive initiative aiming at a future manufacturing paradigm compatible with both the natural environment and human-society. Thus the objectives are oriented towards the environment, the consumer and the manufacturer:

The industrialized economies are fast approaching a period when mass production and mass consumption will be unable to maintain economic growth and increasing wealth. Therefore, there is a need for a manufacturing paradigm shift that will permit a transition from mass production to post one’s patterns of economic activity. Among the salient points, the following are representative:

- Modern engineering can only produce the most circumscribed solutions to the destruction of the global environment, and manufacturing engineering displays an ignorance of global issues.
- Intellectual work is becoming an important component of the industrialized world. The nature of production work is changing and more emphasis is placed on cognitive tasks as opposed to manual work.
- Knowledge is quickly becoming the one most important resource in production systems. Manufacturing companies are selling not only physical products, but also knowledge and information.
- Greater flexibility is required due to higher renewal rates of products, shorter product life-cycles, reduced delivery time, increased customization of products, and JIT philosophies.
- Companies which are structured according to conventional principles are no longer able to respond adequately to change.

The ultimate goal of GNOSIS is to establish a framework for a new manufacturing paradigm which will overcome or minimize the problems inherent in the existing mass production paradigm. Thus a post mass production paradigm (PMPP) is proposed. This involves a new approach to manufacturing, recognizing resource limitations and the balance of nature in order to achieve a sustainable manufacturing environment. The new paradigm will be realized by the manufacture of soft products, with their associated production systems and industrial enterprises. Softness here refers to adaptability, robustness, and growth potential together with congeniality to the natural environment and human society. The lack of such softness in conventional manufactured products is due largely to the uncoordinated use of knowledge, resulting in conceptual blind-spots—local optimization but global inconsistencies. Hence, the effective use of knowledge is regarded as the key to the establishment of the new paradigm.

It is knowledge systematization which will make possible these soft products, soft artifacts and soft manufacturing. Knowledge systematization includes the classification, structuring and organization of knowledge within a systematic framework, giving consideration to the dynamic aspects of knowledge, and the creation of shared ontologies. In this way, the relevant
enabling technologies will be built up availing of knowledge systematization techniques, and knowledge deployment will permit the soft artifacts and virtual manufacturing to be implemented.

The principal objectives of GNOSIS can be outlined as follows:

(1) Environmental Objectives: Modern lifestyles with their reliance on mass manufacturing and mass consumption have caused environmental problems which are gradually forcing a reappraisal of the products and processes supporting the industrialized world. Society is being compelled to move towards the adoption of products which are easily maintained, upgradable, and designed for recycling. The realization of soft products will reduce the volume of manufacturing from today’s levels, thus reducing material usage and pollutant by-products and making manufacturing more environmentally friendly. In manufacturing systems the use of upgradable soft machines will also directly reduce the material consumption and production associated with supplying conventional production machinery which is scrapped when outdated. To be upgradable, such soft machines need to be modular and designed to be re-configurable and upgradable.

(2) Consumer-Oriented Objectives: The human user or consumer of manufactured products is a powerful influence in manufacturing. The consumer drives the rapid introduction of new models, a wider range of equipment options and styling, and the market-place demands for value and price. It is also the underlying reason why “agile manufacturing” has become necessary. What we see is the first stage of a trend towards making products more human-friendly in all respects. The second stage, which is just beginning, will increasingly focus on making them customizable and user-adaptable. To satisfy this demand for human-friendly customization and the movement in manufacturing towards the “batch size of one”, manufacturing systems will have to be dynamically re-configurable to suit rapidly changing production needs. The realization of such systems will involve the implementation of virtual factory concepts and intelligent network implementations. These systems will permit geographically distributed production facilities small in size compared to today’s often massively concentrated production.

(3) Manufacturer-Oriented Objectives: The role of the manufacturer is also central to any research into future products and production. The manufacturer aims at satisfying customer requirements for the lowest possible price and desired quality of goods. The objective of staying at the competitive edge in the manufacturing domain requires that new technologies and research ideas be implemented at the earliest possible stage. Adaptable, upgradable products, machines and manufacturing systems can respond effectively to this pressure.

Thus it can be seen that soft manufacturing involves more than being environmentally friendly. It also implies manufacturing which is sufficiently adaptable to respond to human needs for variety, customization, performance and price. It means human-society-responsive manufacturing in another sense as well. If the production facilities become geographically distributed networks with many of the units being smaller than at present, this could be seen as being both “human friendly” as well as “environmentally friendly”, bringing to reality the adage “small is beautiful”.

3.2 Expected Major Results and Advances in the State-of-the-Art

The principal themes of the research were selected to enable the objectives outlined above and the strategies proposed to be implemented at various levels, so as to achieve results in the short to medium, as well as the long-term. Research into the PMPP goal will include conceptual themes such as social needs, manufacturing philosophy, and the economy, in addition to considering the driving forces and obstacles to it. Research into new competitive soft artifacts and virtual manufacturing, which will form the backbone of the future manufacturing paradigm, will be supported by knowledge systematization and a knowledge intensive engineering framework, with enabling technologies and integration themes focusing on specific issues.

Expected results include the development of enabling tools and technologies for the design and manufacture of new soft artifacts, and also for the achievement of virtual manufacturing. Various examples of soft artifacts will be produced and tested, in the short to medium term. This work will provide a test bed for the full scale production of soft products. The virtual manufacturing research will also be tested and demonstrated in experimental factories before being extended to actual production. Work on the PMPP will provide a blue-print for the transition to the new paradigm, while the knowledge systematization research will form the basis of a “manufacturing knowledge highway” on top of the existing information highway.

In summary, this research proposes a shift from mass material use in production to mass knowledge application, from quantitative to qualitative satisfaction, and the widening of the scope of manufacturing to include the whole product life cycle. The consortium approach towards realizing that goal is by small concrete steps accomplished in technical tasks focused on concrete goals and deadlines.

3.3 Scope and International Dimensions

The scope of the GNOSIS Project covers all aspects of manufacturing: the product, production and the enterprise; the human user or consumer; and the natural environment. In order to avoid any tendency towards a dilution of effort due to this wide scope, the project is concentrated on the attainment of new competitive, environmentally friendly products and production, using knowledge technologies as a foundation.

The rationale for carrying out the proposed research at an international level, involving partners from all the major industrial economies of the world is as follows:

(1) The environmental aspects of manufacturing and consumption cannot be tackled at a local or regional scale, since the environmental consequences of manufacturing and mass consumption are global in scale and necessitate a global approach toward resolving them. Without a world-wide consorted strategy, a rational sustainable balance between manufacturing and the environment cannot be achieved.

(2) The effective use of knowledge as a foundation for tackling world-wide problems requires access to all available knowledge sources. Manufacturing knowledge, environmental knowledge and global market knowledge are not concentrated in any particular region or country but are distributed throughout the world. Thus a research program which utilizes such knowledge must have access to it wherever it is located, and therefore must involve all concerned countries.

(3) Manufacturing is no longer a national or regional concern. The gradual opening of national markets and the international pressure to open up closed markets, the strengthening of world trade bodies, and the consumer desire for the free flow of
products and services, require that a global scope be adopted for any development of new products and production technologies.

(4) The recent advances in communication technologies, as evidenced by advanced electronic networks, multi-media transfer of information and knowledge, and the internationalization of research, make global-level collaborative research feasible for the first time. Such an opportunity should be utilized to advance the living standards and knowledge level of human society.

The GNOSIS Project has partners with experience of environmental issues and market conditions from all the main industrial regions of the globe. Furthermore, they have access to the important knowledge sources for achieving concrete results in the research. Extensive use of state-of-the-art communication technologies is facilitating progress in the project, and the extra value added by collaborating at an international level is ensuring globally balanced, pragmatic, technological advances.

3.4 Project Overview and Approach

The long-term goal of the GNOSIS project is to develop a new manufacturing paradigm which recognizes problems of the present manufacturing environment -- the growing scarcity of natural resources, the problem of environmental destruction, and the issues arising out of regional trade imbalances. The new PMPP is based on systematization of design and manufacturing knowledge to acquire and organize knowledge in a form that supports the design and manufacturing of soft artifacts, i.e., products and factories which achieve reduced resource utilization and waste elimination throughout the whole life cycle from design to recycling or disposal. Major technologies to be investigated include configuration management systems supporting the reuse of engineering and manufacturing knowledge in routine design, and configurable production systems achieving dynamic product-specific manufacturing in flexible production systems. The major characteristics, critical drivers, and obstacles to the PMPP will be studied to guide the work towards the new paradigm.

Electronic communication media will be exploited in order to minimize travel overheads, while ensuring mutual understanding and effective coordination on the research themes. Electronic networks set up during the Test Case will be further expanded. The sharing of methodologies and tools between partners will be continued and inter-regional experiments further developed. Knowledge systematization throughout a complete product life cycle will form an important part of the work.

The consortium approach towards realizing that goal is to have a project structure consisting of 5 linked work packages, each composed of several tasks. Each task will have a sharp technical and industrial focus and limited scope. Some of the tasks will develop tools, introducing softness methodologies and knowledge-oriented strategies, and providing enabling technologies. Others will address the concepts and models that are needed in order to implement the envisioned shift in production paradigm. The number and content of the tasks will evolve over the project duration, as old goals are reached or proven irrelevant and as new goals emerge.

3.5 Technical Themes

The themes of GNOSIS can be summarized as the development and integration of key enabling technology for the improvement of products and production. This will be achieved through the deployment and capitalization of systematized knowledge in order to move
towards a future PMPP. The project is structured in 5 interrelated work packages, corresponding to the technical themes. The relationships between the themes are shown in Figure 1:

![Figure 1 Principal Themes of GNOSIS.](image)

4 PROJECT WORKPLAN

4.1 International Cooperation

The international dimensions have been outlined in section 3.3, emphasizing the rationale behind the global scope of the GNOSIS project. The environmental aspects, the use of knowledge as a fundamental resource, the globalization of markets and the possibilities provided by advances in electronic communication support the ideas behind the GNOSIS effort. Cooperation between the major industrial regions, Japan, Europe and North America exceeded expectations during the Test Case, both as regards input of human and other resources, and as regards the results obtained. The GNOSIS project will continue to build on this international cooperation, with work package participation by all the major regions, as shown in the next section.

4.2 Work Packages, Description and Resource Provisions

The GNOSIS project is organized into a set of 5 conceptually connected, but otherwise autonomous, broad work packages that provide top-down overall control to ensure convergence of the research goals. These work packages consist of more concise, shorter term and technically focused tasks that provide the technical basis, the enabling technology and the deployment actions for the project. The total effort distribution across all regions and work packages is shown in the table 1, after which each of the 5 work packages and their constituent tasks are outlined. Schedules for the tasks to be done in each work package are given for the initial 4 year period, after which further planning can be carried out, as required.
### Table 1 Total Effort Distribution (in Person-Years) over Initial 4 Years

<table>
<thead>
<tr>
<th>Work package</th>
<th>Japan</th>
<th>European Union</th>
<th>North America</th>
<th>Other Regions*</th>
<th>Total Effort (in WPs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>WP1: Post Mass Production Paradigm</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>30</td>
</tr>
<tr>
<td>WP2: Enabling Technologies &amp; Integration</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>57</td>
</tr>
<tr>
<td>WP3: Knowledge Systematisation</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>62</td>
</tr>
<tr>
<td>WP4: Knowledge Deployment 1: Design of Soft Artifacts</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>79</td>
</tr>
<tr>
<td>WP5: Knowledge Deployment 2: Virtual Manufacturing</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>62</td>
</tr>
<tr>
<td>Total Effort (in each Region)</td>
<td>120</td>
<td>124</td>
<td>19</td>
<td>27</td>
<td>290</td>
</tr>
</tbody>
</table>

*Other Regions: EFTA (Switzerland)

---

**Work Package WP1: Post Mass Production Paradigm**

* **Scope and Objectives**

The objective of this work package is to construct the framework for a new manufacturing paradigm that addresses the need to maintain competitiveness in a future consumer society with stringent environmental, market and customer constraints. PMPP research will be used as the global, long-term stimulus for all other activities within GNOSIS. It will provide recommendations for fields of action and supporting tools and methodologies in order to realize artifacts consistent with the PMPP. The knowledge deployment work packages covering soft artifact and virtual factory research will form a key role in the realization of the new paradigm. Topics to be covered will include the definition and evaluation of PMPP-concepts, the identification of key influences at different levels (society, enterprise and shop floor), development of models and tools to run hybrid companies, and development of guidelines to enable fully comprehensive, complete life cycle design of artifacts.

The proposed PMPP will influence society and enterprises, and vice versa, requiring the consideration and investigation of this bi-directional relationship. The rigidity and resistance to change within organizations, both governmental and corporate, are further important factors to be tackled. In the proposed PMPP, companies will have to redesign themselves in order to achieve the required modularity and flexibility.

* **Tasks and Deliverables**

Task 1: Concepts of PMPP
Task 2: Interactions of the PMPP with Enterprise, Environment and Society
Task 3: Softness in the PMPP Context
Deliverables: 1) An in-depth study of guiding principles for future total manufacturing from design to disposal, 2) Recommendations for fields of action for company strategies.

**Work Package WP2: Enabling Technologies and Integration**

- **Scope and Objectives**
  The objective of this work package is to develop the tools and concepts that support other work packages of GNOSIS, namely, the new production paradigm, knowledge deployment and knowledge systematization, and to facilitate their integration in order to reach the overall goals of GNOSIS. Three main complementary areas will be investigated: i) softness methodologies, ii) modeling, and iii) integration.
  
  These areas are explained as follows: 1) Softness methodologies will define and develop the different concepts, principles and technologies required to implement new and powerful functionalities, such as autonomy, adaptability, flexibility and growth potential in any artifact, system or organization. Such softness characteristics are prerequisites for the realization of soft artifacts and soft machines, 2) The objective of modeling is to develop theories, methodologies and prototype implementations to support knowledge deployment in the soft product life cycle, virtual production systems and modeling of soft enterprises, all of which relate to the required characteristics of the future manufacturing paradigm, and 3) The integration research will contribute to the building of distributed intelligent tools to support product life cycle design, the assessment of soft products and enterprises as well as knowledge systematization techniques. The systems to be developed will complement human agent activities in the enterprise, and will ensure the full consistency of all the tasks undertaken in GNOSIS.

- **Tasks and Deliverables**
  
  **Task 1: Softness Methodologies**
  Deliverables: 1) concepts and tools for developing soft artifacts and soft machinery, 2) methods for achieving re-configurable and autonomous production systems, 3) methods and models for non-linear dynamic and chaotic systems to achieve stability and adaptability

  **Task 2: Modeling**
  Tools/methods for life cycle product modeling will be developed, in addition to methodologies for simulating environmental impact. Enterprise modeling will also be studied.
  Deliverables: 1) Product life cycle modeling methods and tools for future manufacturing paradigms, 2) Modeling of enterprise information and decision systems, adapted to permit flexibility, 3) A proposal for hybrid semi-autonomous organizational structures, 4) Demo. and evaluation of the new methods/tools on selected industrial processes/products.

  **Task 3: Integration**
  This task covers specification/implementation of tools for distributed intelligent environments, and develop interface using such tools to coordinate research activities.
  Deliverables: 1) Development of distributed intelligent systems for production systems, 2) Tools for total life cycle modeling, virtual factory, soft products, and other knowledge systematization activities, 3) Demo. of distributed intelligent systems, focusing on organizational/social impacts.
Part Eight  Towards Intelligent Manufacturing Systems

Work Package WP3: Knowledge Systematization

• Scope and Objectives

The aims of this work package are: 1) to apply engineering knowledge more efficiently to engineering tasks, in order to enable the realization of the PMPP. This is called knowledge intensive engineering, 2) to capture, reuse, and update knowledge concerning the complete product life cycle covering design, production, marketing, operation, maintenance, recycling, etc. This is regarded as a prerequisite for the efficient deployment and exploitation of knowledge and for its evolution. The whole process is referred to as “knowledge systematization.”

The objective of this work package is to build a framework on which a new style of engineering activities, called “knowledge intensive engineering,” can take place. Knowledge intensive engineering puts emphasis on more intelligent ways of using engineering knowledge in order to realize soft artifacts and knowledge based manufacturing enterprises. The framework developed in this work package will consist of methodologies, tools and techniques for knowledge intensive engineering, as well as guidelines for their practical implementation. The whole process of developing the framework will be supported and guided by research and experimentation involving organizational, environmental and economical aspects, as well as existing and new technologies. The work package gives a knowledge intensive infrastructure for the knowledge deployment work packages, WP4 and WP5, by using technologies supplied from the integration/enabling technology work package, WP2.

Domains covered will include: design rationale, deep design knowledge, manufacturing knowledge, and other knowledge for product life cycles. Technological, organizational, and methodological aspects will be considered.

Topics will cover: knowledge modeling, knowledge capturing, dynamic knowledge systematization, integrated knowledge management, knowledge encapsulation, integration of knowledge based deployment tools, reasoning techniques, knowledge sharing and reuse, inter-model communications (STEP and other models), knowledge exploitation, knowledge analysis and abstraction, knowledge maintenance, knowledge discovery, dynamic navigation of knowledge use, knowledge media. In the evaluation tasks, different groups of partners will focus on one aspect of one topic in one domain.

• Tasks and Deliverables

Task 1: Establishment of Knowledge Intensive Engineering Framework; This task will involve a survey of current knowledge application practices and identification of needs for the establishment of a knowledge intensive engineering framework (KIEF).
Deliverable: Requirements analysis report for KIEF.

Task 2: Techniques and Organizations for Capturing Knowledge; This task will cover existing and future strategies for acquiring, representing and systematizing knowledge, in order to facilitate its reuse and transferability.
Deliverables: 1) Guidelines for improved knowledge management, 2) Guidelines for where, when, and how to capture/systematize knowledge, 3) Development of software prototypes of knowledge tools employed.

Task 3: KIEF Development and Integration; Development of KIEF, together with its integration with other GNOSIS work. Experimentation will be carried out using the
knowledge deployment work packages to determine requirements for further development. The results of the enabling technologies work package will be utilized.
Deliverables: 1) Reports on KIEF architecture, methodology and functional specification of required technological development, 2) Development of prototypes of KIEF.

Task 4: Application of KIEF; The KIEF framework developed will be applied to various problems in the other GNOSIS work packages, particularly those dealing with knowledge deployment, in order to carry out detailed evaluations.
Deliverables: 1) Reports concerning experimental evaluations, 2) Reports on current practices, organizational and corporate strategies for knowledge use. Guidelines for implementation of KIEF in various business sectors.

Work Package WP4: Knowledge Deployment 1: Design of Soft Artifacts

• Scope and Objectives
The scope of this work package covers theories, methods and tools for applying systematized knowledge in order to realize soft artifacts and move towards the post pass production paradigm. Results from the enabling technologies and knowledge systematization work packages will be used to develop and implement tools. Requirements concerning knowledge deployment in design will be fed back to these work packages and will also provide input to the virtual manufacturing work package.

To achieve minimum use of natural resources and the minimum impact on the environment, soft artifacts must be closely matched to requirements as they will have many more variants with increased one-off or low-volume production than conventional products. In addition, the design and configuration of the artifact will have to take into account all life cycle stages, e.g., usage, maintenance, re-configuration, reuse and disposal.

• Tasks and Deliverables
Task 1: Design methodologies
Deliverables: Generic design methodologies in order to attain softness in artifacts, including modularity, self-organization, autonomy, reconfigurability and self-maintainability.

Task 2: Configuration Methods
Deliverables: Design configuration tools

Task 3: Examples of Soft Artifacts:
Deliverables: 1) Examples of soft artifacts including both software and hardware, 2) Evaluation of tools and methodologies

Work Package WP5: Knowledge Deployment 2: Virtual Manufacturing

• Scope and Objectives
The objective of this work package is to promote the transition from the current manufacturing paradigm to the PMPP by developing and applying virtual factory and manufacturing concepts.

The virtual factory refers to the use of modeling and simulation to enable the integration of all aspects of management of manufacturing. Systematized product and
manufacturing knowledge is used as a fundamental resource for the creation of the manufacturing process and also for its management. This knowledge is combined with soft artifact concepts and the enabling technologies provided by other work packages in order to realize the virtual factory, which in turn will provide feedback to other work packages.

The study and development of reconfigurable tools for management of decentralization and customization in manufacturing will form a key theme in the research. Other specific matters to be addressed include the implementation of distributed architectures and supporting systems, in addition to self-organization concepts and on-line re-configuration and control. The ultimate goal is to provide companies with improved methodologies/tools for more competitive management.

**Tasks and Deliverables**

**Task 1: Knowledge Based Process Design;** Tools and methods for automated process planning based on systematized knowledge for both product/production systems will be developed.

**Deliverables:** 1) Software capable of automatically creating process plans containing information needed by NC-code generators (feature based factory model and manufacturing knowledge), 2) Advanced graphical user interface easing the control of information for process plan creation

**Task 2: Development of simulation technology;** In this task simulation technology is applied to areas such as manufacturing, construction, ergonomics, training and prototyping. Distributed modeling and simulation will also be evaluated from the concurrent engineering perspective.

**Deliverables:** 1) Definition of linkages between factory simulators and real production databases, 2) Definition of data needed for advanced process simulators.

**Task 3: Exploration of emerging scheduling paradigms;** An efficient management of time is critical for the success of a company. Increasing flexibility in the manufacturing processes is continuously raising the degree of freedom in scheduling. This fact will also have implications on the structure of the organizations.

**Deliverables:** 1) Analysis of scheduling requirements in distributed and non-distributed manufacturing environments, 2) Specification and evaluation of scheduling architectures, 3) Implementation of prototypes in both simulated and real factory environment.

**Task 4: Development of architecture for Virtual Factory environment**

Different technologies supporting the virtual factory concept will be integrated in order to address the organization of distributed manufacturing.

**Deliverables:** 1) Specification for interfaces between different virtual factory functionalities and environments, 2) Implementation of results of work-packages in real production processes.

**Task 5: Pilot test cases for the Virtual Factory concept**

The virtual factory concept will be applied and evaluated in pilot cases covering different areas in the manufacturing process: construction, shop floor, assembly.

**Deliverables:** 1) Prototype installations, 2) Benchmarking report
5 CONSORTIUM COMPOSITION

Regions Involved:
Japan, EU, Canada, EFTA, (US participation under negotiation)

International Coordinating Partners:
- Japan Region: Mitsubishi Electric Corporation
- European Union (EU) Region: ADEPA (Agence de la Productique)
- Canada Region: KSI University of Calgary
- EFTA Region: ETH Zurich

Regional Consortium Partners:
- **Japan Region**
  - Coordination Partner: Mitsubishi Electric Corporation
  - Industrial Partners: Mitsubishi Electric, Shimizu, Kajima, Chiyoda, Yamatake Honeywell
  - Academic Partners: University of Tokyo, Nara Advanced Institute of Science and Technology, Osaka University, ASTEM

- **European Union (EU) Region**
  - Coordination Partner: ADEPA (Agence de la Productique)
  - Industrial Partners: ADEPA, IBM France, BICC, ITMI-Aptor, ABB, Tehdasmallit Oy, Becos, Siemens, Schenck Eng., SKET.
  - Academic Partners: Cambridge University, LLP-Cesalp, Helsinki University of Technology, Tampere University of Technology, Ecole des Mines, IAF Magdeburg
  - National Research Institutes: Fraunhofer Institute IPA, Technical Research Center of Finland VTT.

- **EFTA Region**
  - Coordination Partner: Swiss Federal Institute of Technology ETH Zurich
  - Academic Partners: Swiss Federal Institute of Technology EPFL, Swiss Federal Institute of Technology ETHZ.

- **Canada Region**
  - Coordination Partner: KSI University of Calgary
  - Academic Partners: DME University of Calgary, KSI University of Calgary

6 CONCLUSIONS AND ACKNOWLEDGEMENTS

This paper has described the objectives and present situation of the GNOSIS consortium with the international IMS research program. The development of this collaborative research is a joint development with many GNOSIS consortium colleagues, especially, Prof. T.Tomiyama (University of Tokyo, Japan), Prof. B.R.Gaines (University of Calgary, Canada), Prof. B.Faltings (EPFL, Switzerland) and Prof. M.Mäntylä (HUT, Finland). Please contact the author if you have some interest in GNOSIS. Newcomers interested in the objectives described are welcome to join the GNOSIS consortium.
INDEX OF CONTRIBUTORS

<table>
<thead>
<tr>
<th>Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acebedo, C.M.</td>
<td>371</td>
</tr>
<tr>
<td>Akomode, O.J.</td>
<td>242</td>
</tr>
<tr>
<td>Anderl, R.</td>
<td>47</td>
</tr>
<tr>
<td>Arai, E.</td>
<td>181</td>
</tr>
<tr>
<td>Atsumi, R.</td>
<td>301</td>
</tr>
<tr>
<td>Banerjee, S.K.</td>
<td>81</td>
</tr>
<tr>
<td>Benadjaoud, G.N.</td>
<td>206</td>
</tr>
<tr>
<td>Brand, M.</td>
<td>155</td>
</tr>
<tr>
<td>Bunce, P.</td>
<td>418</td>
</tr>
<tr>
<td>Chrobot, J.</td>
<td>356</td>
</tr>
<tr>
<td>Clements, P.E.</td>
<td>169</td>
</tr>
<tr>
<td>Coutts, I.A.</td>
<td>169</td>
</tr>
<tr>
<td>David, B.T.</td>
<td>206</td>
</tr>
<tr>
<td>Dubois, E.</td>
<td>69</td>
</tr>
<tr>
<td>Edwards, J.M.</td>
<td>169</td>
</tr>
<tr>
<td>Goncharenko, I.</td>
<td>395</td>
</tr>
<tr>
<td>Goossenaerts, J.</td>
<td>193, 371</td>
</tr>
<tr>
<td>Henning, G.P.</td>
<td>103</td>
</tr>
<tr>
<td>Höpf, M.</td>
<td>431</td>
</tr>
<tr>
<td>Irgens, C.</td>
<td>242</td>
</tr>
<tr>
<td>Jäkel, F.-W.</td>
<td>231</td>
</tr>
<tr>
<td>Jansen, M.H.</td>
<td>383</td>
</tr>
<tr>
<td>Janusz, B.</td>
<td>140</td>
</tr>
<tr>
<td>Jordan, J.</td>
<td>418</td>
</tr>
<tr>
<td>Kamio, Y.</td>
<td>409</td>
</tr>
<tr>
<td>Kasashima, N.</td>
<td>395</td>
</tr>
<tr>
<td>Katzy, B.R.</td>
<td>221</td>
</tr>
<tr>
<td>Kimura, F.</td>
<td>277</td>
</tr>
<tr>
<td>Kojima, T.</td>
<td>313</td>
</tr>
<tr>
<td>Kosanke, K.</td>
<td>59, 115</td>
</tr>
<tr>
<td>Kurihara, T.</td>
<td>418</td>
</tr>
<tr>
<td>Lambert, A.J.D.</td>
<td>383</td>
</tr>
<tr>
<td>Lees, B.</td>
<td>242</td>
</tr>
<tr>
<td>Leone, H.P.</td>
<td>103</td>
</tr>
<tr>
<td>Mannarino, G.S.</td>
<td>103</td>
</tr>
<tr>
<td>Martinsen, K.</td>
<td>313</td>
</tr>
<tr>
<td>Mase, T.</td>
<td>3</td>
</tr>
<tr>
<td>Matsuda, M.</td>
<td>277</td>
</tr>
<tr>
<td>McLean, C.R.</td>
<td>20</td>
</tr>
<tr>
<td>Mendonça, J.M.</td>
<td>327</td>
</tr>
<tr>
<td>Mertins, K.</td>
<td>231</td>
</tr>
<tr>
<td>Messina, S.</td>
<td>344</td>
</tr>
<tr>
<td>Mills, J.J.</td>
<td>155</td>
</tr>
<tr>
<td>Mori, K.</td>
<td>395</td>
</tr>
<tr>
<td>Petit, M.</td>
<td>69</td>
</tr>
<tr>
<td>Pinto Ferreira, J.J.</td>
<td>327</td>
</tr>
<tr>
<td>Pleinevaux, P.</td>
<td>344</td>
</tr>
<tr>
<td>Pütter, J.</td>
<td>335</td>
</tr>
<tr>
<td>Rajala, M.M.</td>
<td>254</td>
</tr>
<tr>
<td>Rakowski, J.</td>
<td>356</td>
</tr>
<tr>
<td>Reid, S.P.</td>
<td>81</td>
</tr>
<tr>
<td>Reithofer, W.</td>
<td>128</td>
</tr>
<tr>
<td>Riggers, B.</td>
<td>221</td>
</tr>
<tr>
<td>Schaeffer, C.F.</td>
<td>431</td>
</tr>
<tr>
<td>Schuwer, R.V.</td>
<td>383</td>
</tr>
<tr>
<td>Schwermer, M.</td>
<td>231</td>
</tr>
<tr>
<td>Smalec, Z.</td>
<td>356</td>
</tr>
<tr>
<td>Splinter, M.A.M.</td>
<td>383</td>
</tr>
<tr>
<td>Stamminger, M.</td>
<td>265</td>
</tr>
<tr>
<td>Takata, M.</td>
<td>181</td>
</tr>
<tr>
<td>Toyama, M.</td>
<td>439</td>
</tr>
<tr>
<td>Uhl, J.</td>
<td>356</td>
</tr>
<tr>
<td>van Bommel, P.J.M.</td>
<td>32</td>
</tr>
<tr>
<td>van der Hoeven, A.</td>
<td>289</td>
</tr>
<tr>
<td>van Leuken, G.R.</td>
<td>289</td>
</tr>
<tr>
<td>Vernadat, F.B.</td>
<td>59</td>
</tr>
<tr>
<td>Vosniakos, G.</td>
<td>91</td>
</tr>
<tr>
<td>Wasmer, A.</td>
<td>47</td>
</tr>
<tr>
<td>Williams, M.</td>
<td>91</td>
</tr>
<tr>
<td>Zelm, M.</td>
<td>59</td>
</tr>
<tr>
<td>Zuijdgeest, L.</td>
<td>11</td>
</tr>
<tr>
<td>Keyword Index</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>action research 242</td>
<td></td>
</tr>
<tr>
<td>adaptation methods 356</td>
<td></td>
</tr>
<tr>
<td>advanced manufacturing systems 418</td>
<td></td>
</tr>
<tr>
<td>agility 418</td>
<td></td>
</tr>
<tr>
<td>anti-positivism 242</td>
<td></td>
</tr>
<tr>
<td>application protocol 47</td>
<td></td>
</tr>
<tr>
<td>architecture 32, 431</td>
<td></td>
</tr>
<tr>
<td>artifact life phase service body 193</td>
<td></td>
</tr>
<tr>
<td>assessment 254</td>
<td></td>
</tr>
<tr>
<td>business modelling 115</td>
<td></td>
</tr>
<tr>
<td>business process 231, 254, 301</td>
<td></td>
</tr>
<tr>
<td>business process analysis 231</td>
<td></td>
</tr>
<tr>
<td>management 265</td>
<td></td>
</tr>
<tr>
<td>modelling 59, 231</td>
<td></td>
</tr>
<tr>
<td>reengineering 69, 231, 265</td>
<td></td>
</tr>
<tr>
<td>CAD frameworks 289</td>
<td></td>
</tr>
<tr>
<td>CAD/CAM 3</td>
<td></td>
</tr>
<tr>
<td>CALS 11, 193</td>
<td></td>
</tr>
<tr>
<td>certification 231</td>
<td></td>
</tr>
<tr>
<td>Computer Integrated Manufacturing (CIM) 103, 206, 335</td>
<td></td>
</tr>
<tr>
<td>CIMOSA 59, 128, 140, 221</td>
<td></td>
</tr>
<tr>
<td>Coad-Yourdon’s OOA 91</td>
<td></td>
</tr>
<tr>
<td>communication protocols 335</td>
<td></td>
</tr>
<tr>
<td>computer networks 335</td>
<td></td>
</tr>
<tr>
<td>concurrent engineering 11</td>
<td></td>
</tr>
<tr>
<td>control system 356, 431</td>
<td></td>
</tr>
<tr>
<td>data exchanges 206</td>
<td></td>
</tr>
<tr>
<td>data modelling 206, 289</td>
<td></td>
</tr>
<tr>
<td>databases 206</td>
<td></td>
</tr>
<tr>
<td>deployment 301</td>
<td></td>
</tr>
<tr>
<td>design 254, 301</td>
<td></td>
</tr>
<tr>
<td>design flow management 289</td>
<td></td>
</tr>
<tr>
<td>digital process 3</td>
<td></td>
</tr>
<tr>
<td>distributed programming/execution environment 181</td>
<td></td>
</tr>
<tr>
<td>EDI 3</td>
<td></td>
</tr>
<tr>
<td>enabling technology 439</td>
<td></td>
</tr>
<tr>
<td>enterprise architectures 115</td>
<td></td>
</tr>
<tr>
<td>enterprise integration 59, 103, 115, 221, 327, 371, 409</td>
<td></td>
</tr>
<tr>
<td>enterprise modelling 59, 81, 103, 115, 128, 140, 231</td>
<td></td>
</tr>
<tr>
<td>enterprise reference architectures 327</td>
<td></td>
</tr>
<tr>
<td>environmental information systems 383</td>
<td></td>
</tr>
<tr>
<td>Ethernet 335</td>
<td></td>
</tr>
<tr>
<td>evaluation 301</td>
<td></td>
</tr>
<tr>
<td>event management 344</td>
<td></td>
</tr>
<tr>
<td>executable models 327</td>
<td></td>
</tr>
<tr>
<td>experiences and failures in reengineering 265</td>
<td></td>
</tr>
<tr>
<td>EXPRESS 81, 313</td>
<td></td>
</tr>
<tr>
<td>extended enterprises 155</td>
<td></td>
</tr>
<tr>
<td>factory automation 169</td>
<td></td>
</tr>
<tr>
<td>formal specification language 69</td>
<td></td>
</tr>
<tr>
<td>fractals 418</td>
<td></td>
</tr>
<tr>
<td>framework for enterprise integration 193</td>
<td></td>
</tr>
<tr>
<td>functional and quality requirements 69</td>
<td></td>
</tr>
<tr>
<td>global manufacturing 409</td>
<td></td>
</tr>
<tr>
<td>Globeman 409</td>
<td></td>
</tr>
<tr>
<td>GNOSIS 439</td>
<td></td>
</tr>
<tr>
<td>graphical interface 395</td>
<td></td>
</tr>
<tr>
<td>heterogeneous systems integration 155</td>
<td></td>
</tr>
<tr>
<td>holon 431</td>
<td></td>
</tr>
<tr>
<td>human aspects 32</td>
<td></td>
</tr>
<tr>
<td>human protocol 301</td>
<td></td>
</tr>
<tr>
<td>ICT-infrastructure 221</td>
<td></td>
</tr>
<tr>
<td>IEM 231</td>
<td></td>
</tr>
<tr>
<td>indicator 265</td>
<td></td>
</tr>
<tr>
<td>industrial development 371</td>
<td></td>
</tr>
<tr>
<td>information elements 344</td>
<td></td>
</tr>
<tr>
<td>information infrastructure 81, 193, 371</td>
<td></td>
</tr>
<tr>
<td>information systems 155, 169, 242</td>
<td></td>
</tr>
<tr>
<td>information technology 11, 242</td>
<td></td>
</tr>
<tr>
<td>infrastructural software system 181</td>
<td></td>
</tr>
<tr>
<td>infrastructure 155</td>
<td></td>
</tr>
<tr>
<td>integrating infrastructure 344</td>
<td></td>
</tr>
<tr>
<td>integration 155, 206, 254</td>
<td></td>
</tr>
<tr>
<td>integration infrastructure 169</td>
<td></td>
</tr>
<tr>
<td>integration platform 289</td>
<td></td>
</tr>
<tr>
<td>Intelligent Manufacturing System (IMS) 409, 418, 431, 439</td>
<td></td>
</tr>
<tr>
<td>intelligent systems 418</td>
<td></td>
</tr>
<tr>
<td>internal structure 301</td>
<td></td>
</tr>
<tr>
<td>ISO9000ff 231</td>
<td></td>
</tr>
<tr>
<td>key driven business reengineering 265</td>
<td></td>
</tr>
<tr>
<td>key factor 265</td>
<td></td>
</tr>
<tr>
<td>knowledge capitalization 439</td>
<td></td>
</tr>
<tr>
<td>knowledge elicitation 242</td>
<td></td>
</tr>
<tr>
<td>knowledge highway 439</td>
<td></td>
</tr>
<tr>
<td>knowledge intensive engineering 439</td>
<td></td>
</tr>
<tr>
<td>knowledge systematization 439</td>
<td></td>
</tr>
<tr>
<td>life cycle 327, 439</td>
<td></td>
</tr>
<tr>
<td>life cycle modelling 193</td>
<td></td>
</tr>
<tr>
<td>logistic support 11</td>
<td></td>
</tr>
<tr>
<td>logistics 221</td>
<td></td>
</tr>
<tr>
<td>low-cost 356</td>
<td></td>
</tr>
</tbody>
</table>
Keyword Index

machine tool 395
machining preparation 277
manufacturing 155, 221, 431, 439
manufacturing engineering 20
manufacturing process modelling 344
manufacturing software integration 20
manufacturing systems 91
manufacturing work-cell control system 181
mapping 47
mapping language 47
mapping table 47
milling machine model 277
MO2GO 231
modelling 221, 327, 439
modelling languages 115
modelling methodologies 115
monitoring 395
NC data generation 277
network 221
new paradigm 439
Next Generation Manufacturing Systems (NGMS) 418
object oriented analysis 103
object oriented model 206
object-oriented 231
objects 344
PDM 3
physical flow information 383
planning 301
process modelling 20
process optimization 140
process re-engineering 221
process redesigning 140
process/operation planning 277
product life cycle 409
product model 277
product modelling 313
QM-documentation 231
reengineering quality manual 231
risks 242
SGML 3, 11
shop floor 91
shop floor control 32
shop floor control system 356
simulation 20, 128, 254
small and medium size enterprises 371
soft artifacts 439
software engineering 356
STEP 3, 11, 81, 313
strategy 221
sustainable production 383
system analysis 91
system development approach 32
systems integration architecture 155
target system 265
TELEflow 221
three-schema 169
unit of functionality 47
user interface 289
value analysis 254
value system 221
variation 254
vectorial tolerancing 313
vehicle CALS 3
view 47
virtual 221
virtual enterprise 155, 409
virtual manufacturing 20, 277, 439
virtual reality 395
workflow 59
World Wide Web 395