16346-E

Distr.
LIMITED

IPCT.63
17 May 1988

ORIGINAL: ENGLISH

UNITED NATIONS
INDUSTRIAL DEVELOPMENT ORGANIZATION

SOFTWARE PRODUCTION: ORGANISATION AND MODALITIES*

Prepared by

Hans-Jochen Schneider**

UNIDO Consultant

---

* The views expressed in this document are those of the author and do not necessarily reflect the views of the secretariat of UNIDO. This document has been reproduced without formal editing.

** Professor of Computer Science, Technical University of Berlin; President, ACTIS zentrale Verwaltungs-GmbH, Stuttgart, Federal Republic of Germany.

V.88-24831

Explanatory notes

Reference to "dollars" ($) are to United States dollars.

Mention of firm names and commercial products does not imply the endorsement of UNIDO.

Summary

Software production is emerging as one of the most important services in this age of information-driven technology. This makes it crucial to examine the organisation and modalities of software production, especially as it applies to developing countries. Importing software that is unstructured and polluted from industrialised countries has to be handled with the utmost care and consideration.

The study describes the great potential of software development in developing countries through North-South co-operation, based on the encouraging success of Singapore. It discusses new techniques of software production in so-called software production environments, along the lines of Japan's fifth generation computer project. Drawing on his experiences as the co-founder of a German software house, the author presents suggestions on installing a software house in developing countries and considers the economic issues.

The major contribution of the report is a series of recommendations for promotional action on software development. The idea for a Technology Transfer Centre and an international Centre for Microelectronic Applications and Software (MAS) is explored, including a range of activities that can be undertaken.

CONTENTS

CONTENTS (continued)

LIST OF FIGURES

Introduction

Microcomputers have a low energy consumption and do not pollute our world. Software systems on microcomputers have shown enormous problem-solving power and capabilities. Industrialised societies are being transformed into information and services societies. Information is getting a production factor. It is therefore extremely important that developing countries develop national policies for introducing hardware distribution and software production capabilities.

Computer science experts in developed countries are rare and command a high price. Education standards in developing countries are rather high. These factors present a great opportunity to build up a fruitful North-South cooperation in software trade and production. Developed country experience with industrialised-oriented software production has had a relatively brief history. Therefore, a careful penetration into this market segment seems to be feasible for developing countries.

Techniques and tools in computer science are developing fast and continuous monitoring of developments is necessary. Personnel costs will continue to increase in the future, so it is important to avoid individualised programming and to develop industry-oriented techniques and methods for software production. Software production units, often called software houses, have to be built up like other industrial companies.

I. ANALYSIS OF SOFTWARE PRODUCTION PROCESSES IN DEVELOPED COUNTRIES

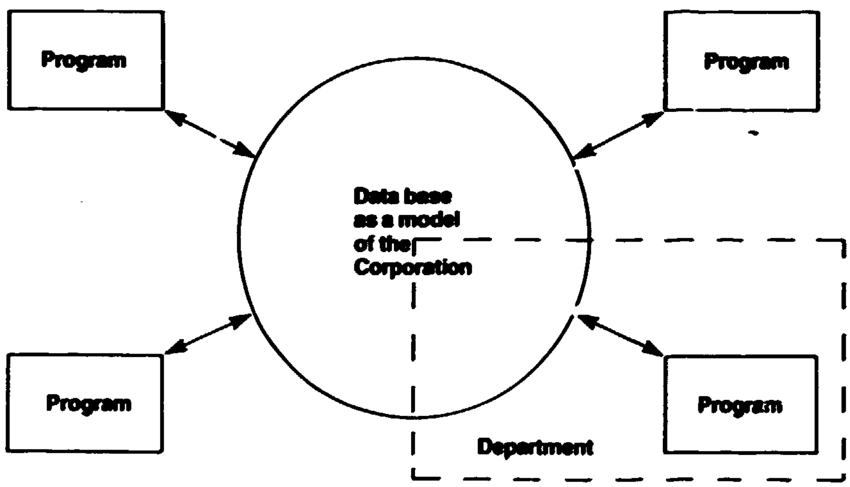
In building large software systems and complex information systems, we are addressing a semantic realm and a complexity which cannot be covered by one known technique or by one set of tools. We have to bring our theoretical knowledge and our practical and engineering-oriented experience together to be more successful in solving these complex problems.

Results from complexity theory show that decision algorithms grow superexponentially even for trivial problems. We need a construction primer, like architects and engineers. We have to be able to predict the behaviour of our information systems before implementing them. We have to be able to plan them in advance, and we have to look at the consequences from the technological point of view.

Since we can prove the correctness of a program according to its formal specifications only for small problems, we have to support the specification definition process. If the specification of a solution is wrong, then the correctness proof is useless. The program was the center of gravity in the last 30 years of data processing. This was a Ptolemaic conception of the world of data processing (Fig. 1a). Since 1970, our understanding towards a Copernican view of data processing (Fig. 1b) has been developing.

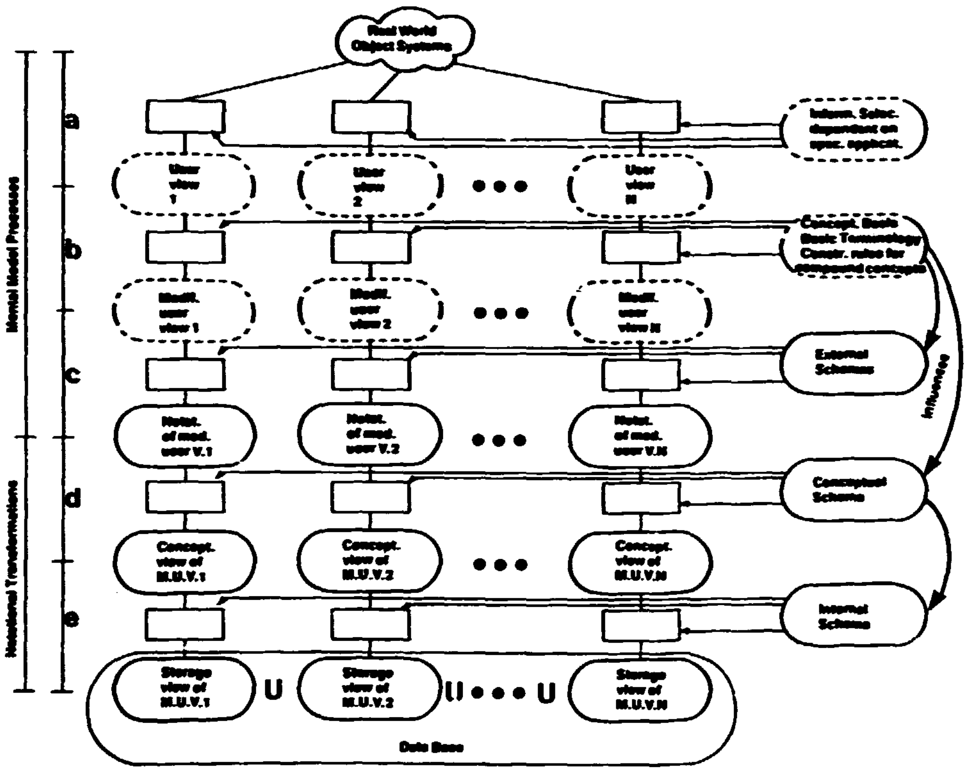
Fig. 1a: "PTOLEMAIC" conception of the world of data processing

Fig. 1b: "COPERNICAN" conception of the world of data processing



One step towards building more consistent systems was the development of data base management systems. These systems allowed many users to have simultaneous access to big data bases. Up to 1000 application programs are built upon such a data base. Each program can, theoretically, have a different logical view of the physical data base. The application programs are data-independent. Physical reconstruction of the data base does not influence the application program. In modern data base management systems, mapping parts of the real world into a data base follows the conception of Fig. 2.

Fig. 2: Mapping parts of the real world into a data base (coexistence model)



After a filter process of information selection dependent on a specific application, each user has an individual mental user view. An essential prerequisite for people to understand each other is a conceptual basis, a basic terminology and construction rules with which to construct compound concepts. This yields to so-called modified user views, now semantically compatible. In a man-machine-communication process, these prerequisites have to be stated formally in external and conceptual schemes according to the three-level systems architecture of ANSI (American National Standards Institute). After the modified user view has been stated in notations, the notations are mapped through several levels down to the physical data base.

Besides the development of generalised data base management systems, big specialised worldwide on-line systems have been built. A skeleton of frequencies is tabulated in Fig. 3.

Fig. 3: Skeleton of frequencies of information systems

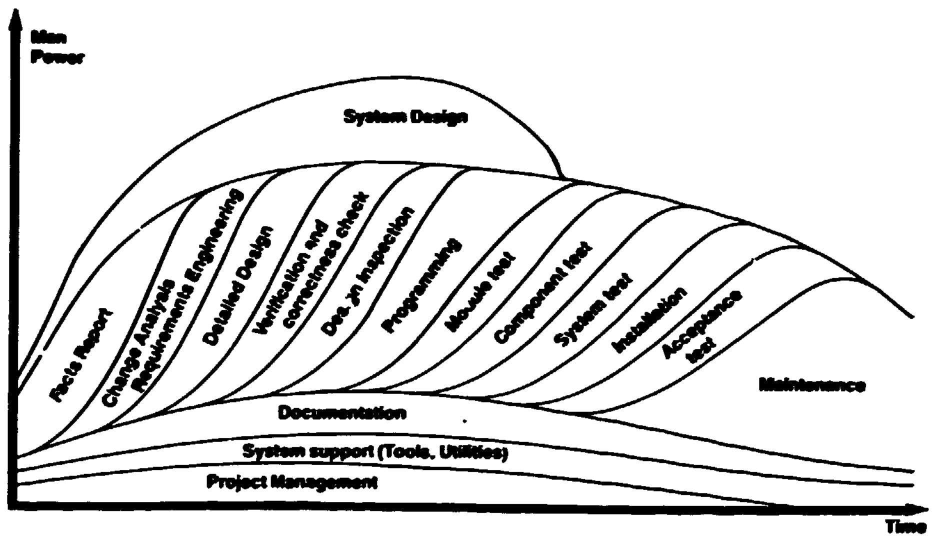
	UNITED AIRLINES	AMERICAN AIRLINES	PAN AMERICAN AIRLINES	HERTZ (rent a car)	SHERATON (Hotel)
STORED DATA (MBYTES)	6 000	12 000	1 700	1 600	39
TRANSACTIONS (MILLION PER YEAR)	1 800	2 200	1 800	64	14,5
COMPUTER	370/195 (2x) 370/65	AMDAHL 470	370/158	UNIVAC 1182 (2x)	
TERMINALS	6 000	6 000	1 500	3 000	400
PERSONNEL	MAINTENANCE: 65 OPERATIONS: 66	MAINTENANCE: 75 OPERATIONS: 40	MAINTENANCE: SW.13, HW.40 OPERATIONS: 12		
SOFTWARECOSTS DESIGN + Implement (MILL. US$)	25 - 35		1,2 and 40 Manyears	183 Manyears	8,6
Hardwarecosts (MILL. US$ PER YEAR)	10			3 (only for Telecommunication)	4

These developments have shown that we have reached our capability threshold of handling complex problems. Results of complexity theory say that the number of decision algorithm steps, even for trivial problems, grows superexponentially (/RA74/) and that we can prove correctness only for tiny problems.

Also from the practical side, we know that big operating systems and data base management systems contain 150-300 errors. Today we realise that the first steps of the problem-solving process were fully underestimated, that our systems architecture did not support evolutionary systems and that we have not developed enough tools to support the different phases of systems development (see Fig. 4). DAVIS and VICK (/Da77/) estimate a 70 per cent failure rate in the first phases of the problem solving process.

Limitations for algorithmic modelling and a discussion on how to cope with complexity can be found in /Ch80a/,/Scn76/ and /Scn80/.

Fig. 4: Qualitative person power estimates for the different phases of the systems development process



The facts imply that techniques and tools for information system development (see /Scn79/) are needed urgently. Fig. 5 outlines criteria for evaluation of information systems specification, realisation, information systems and tools. This list can by no means be considered complete, but it does reflect prerequisites for a conceptual basis and the corresponding languages.

Fig. 5: Criteria for the evaluation of information systems specification concepts and languages

* INDEPENDENCE OF THE PROBLEM STRUCTURE
* INDEPENDENCE OF THE PROBLEM SIZE
* INDEPENDENCE OF THE DATA PROCESSING CONCEPTS AND TOOLS
* COMPLETENESS OF CONCEPTS
  (ACTIVITIES, DATA STRUCTURES, PRECEDENCE, CONTROL FLOW, DATA FLOW)
* STRUCTURAL POWER
  (TREE, ACYCLIC GRAPHS, GENERAL GRAPHS)
* APPLICABILITY TO ORGANISATION AND/OR DP
* EASE OF LEARNING AND TEACHING
* ABSTRACTION/REFINEMENT-HOMOMORPHISMS ON THE ACTIVITY AND/OR THE INFORMATION SIDE
* TERMINOLOGICAL CONTROL, SEMANTICS FORMALISATION, REPRESENTATION OF KNOWLEDGE
* FLEXIBILITY IN CHANGES
* SUPPORT OF THE WHOLE LIFE CYCLE OF PROBLEM-SOLVING
* THEORETICAL BASIS EXISTING

In this subject area, many formal methods have been applied. Interested readers are referred to the following papers: /Go80/, /Go80a/, /Ham79/, /Li79/, /Me75/, /Mu79/, /Sce80/, /Se80/. Other information systems development methodologies are described in /Al78/, /Ja77/, /Sa82/, /Scn82a/, /Scn82b/, /Scn82c/.

The following theses describe the situation of the data processing market in developed countries:

Thesis 1: More and more companies utilise data processing internally and in their products. Therefore data processing is a matter of concern to top management.

Thesis 2: The industrial society is in transition towards an information society. Information has to be evaluated and consequently introduced in balance sheets.

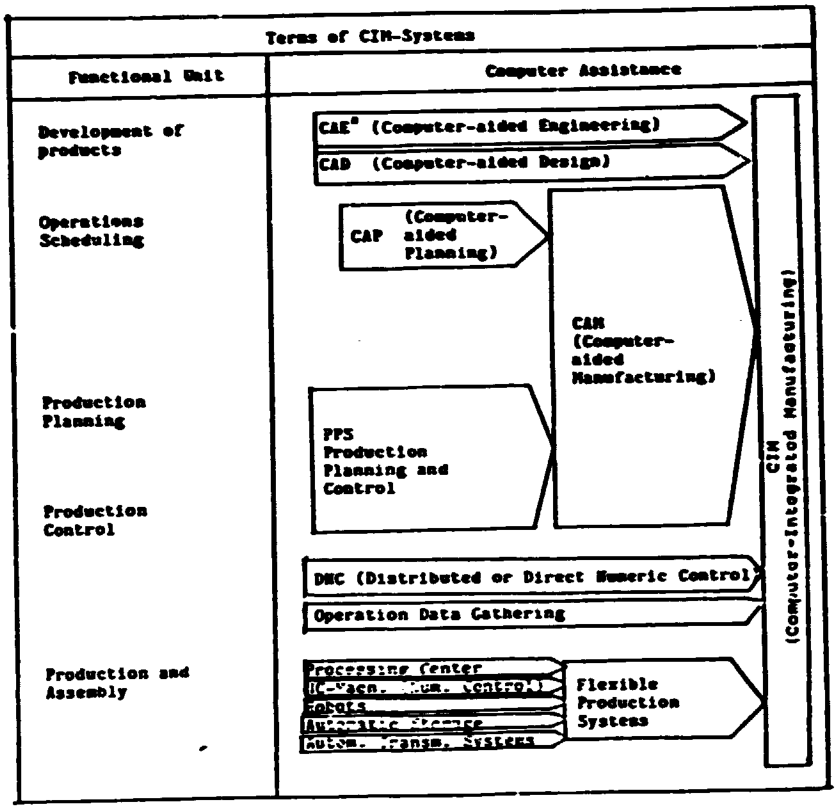
Thesis 3: The speed of acting on the market will be the crucial measure of competence of a company in the future. The swiftness of information gathering, processing and transferring is an unconditional prerequisite for the secured existence of a company.

Thesis 4: All functional units of a company will be part of an integrated company information system. Decentralised organisation and data processing will be linked with centralised information management.

Thesis 5: Software, not hardware, has become the decisive cost and evaluation factor when introducing and following up data processing solutions.

Besides office automation, the area of computer-integrated manufacturing (CIM) will be of greatest interest for the future. One of the major developments of the past five years in data processing is the integration of commercial data processing, text processing, engineering data processing (computer-aided design, computer-aided manufacturing, computer-aided engineering, or CAD, CAM, CAE), process control and telecommunication. As a consequence, the various information islands in the company are linked together, yielding computer-integrated manufacturing of the third and fourth generations (see Fig. 6). All information units concerning order entry, product design, analysis and simulation, process control, quality assurance, material handling, assembly inspection, test, inventory control and shipping are handled by the same data base.

Fig. 6: Information islands are linked together



It is essential for companies to create the position of information manager. The manager should be responsible for the whole information flow within the company and for introducing concepts and tools of information technology in the company products. The manager must be part of top management and must head the information center in which all information about the company, its products, the market, existing and future technologies and legislation flow together. Furthermore, the information manager must co-ordinate the co-operation of the various departments, consultants, organisational people, staff union, revisers and staff responsible for data protection.

The available information allows an exact charting of the company's strategic and tactical moves and the simulation of alternative decision strategies. Existing methods and software tools from econometrics, operations research and system dynamics must be extended and developed.

II. TRENDS IN THE SOFTWARE MARKET IN DEVELOPED COUNTRIES

The following trends are predicted by a London market survey company, INPUT Ltd. A $48 billion information services market by 1991 is predicted for Western Europe.

* The overall market for information services in the four major country markets of Western Europe (France, Italy, the United Kingdom and the Federal Republic of Germany) exceeded $18 billion in 1986.

* This market is estimated to grow at an average annual growth rate of 21 per cent to reach $48 billion by 1991.

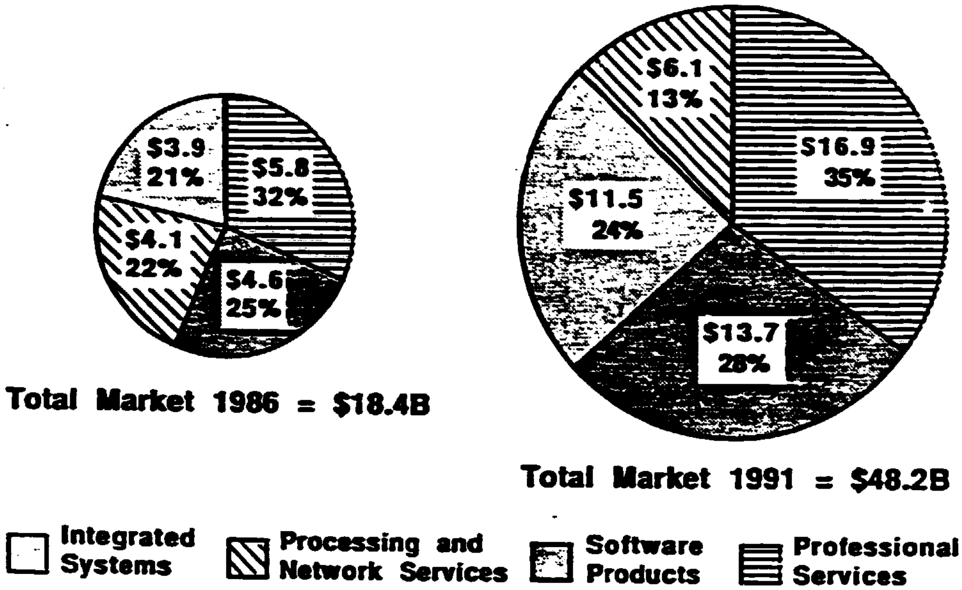
* The highest levels of growth will be in the software products and professional services sectors: 24 per cent average annual growth rate. Each will increase its absolute share of the market, largely at the expense of processing and network services.

* The integrated systems sector is expected to grow at the slightly lower annual growth rate of 23 per cent to represent nearly a quarter of the market in 1991 (up from 21 per cent in 1985).

* Processing and network services is expected to achieve only 8 per cent annual average growth to 1991. Thus, this sector, representing nearly 25 per cent of the 1985 market, will fall to a 13 per cent share of the 1991 market.

* Each of these major industry sectors is discussed in more detail below. A pie chart (Fig. 7) shows the total market in 1986 and the predicted one for 1991.

Fig.7: $48 billion services market by 1991 (Western Europe)



Processing and network services market growth

* INPUT forecasts that the processing and network services sector will achieve an average annual growth rate of 8 per cent, expanding revenues from $4.1 billion in 1986 to $6.1 billion by 1991 (see Fig. 8).

* The most significant driving force for growth during this five-year forecast period is the convergence of computing and telecommunications technology.

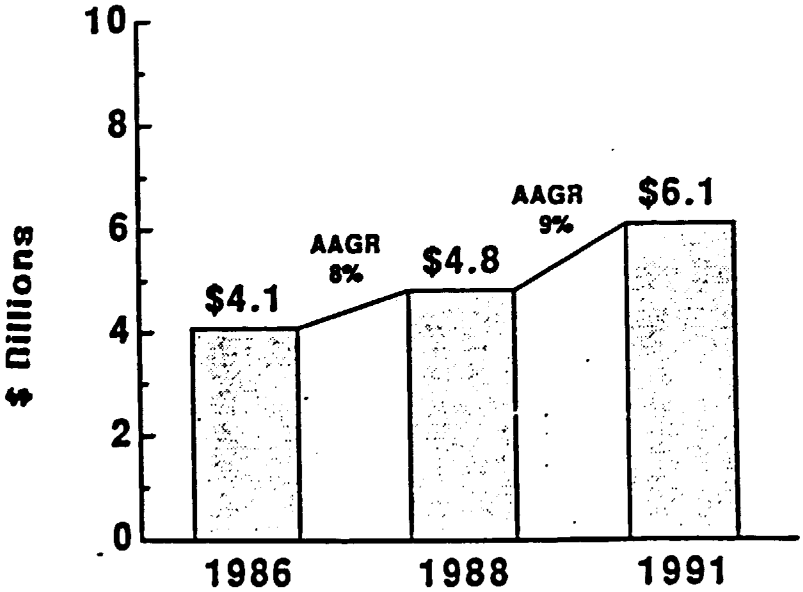
* The development of value-added network services (VANS) is providing major new opportunities for services vendors in the increasingly liberalised Western European telecommunications environments.

* Processing facilities management and network facilities management are emerging as important opportunities. The market is being driven by new requirements for increasingly complex service solutions. The provision of professional services and software can far outweigh the processing element in the service.

* Processing and network service vendors must refocus their strategic thrust towards market niches in specialist application areas. In addition, it is important for vendors to develop a total service orientation that augments basic processing services with the provision of specialist application software and consultancy expertise.

* Trends towards the development of departmental systems are providing total service opportunities as vendors can leverage their knowledge of specialised markets in combination with expertise in the provision of networked systems.

Fig.8: Processing and network services market growth (average annual growth rate)



Network service directions

* The major impact of computer/communications convergence on the information services industry has been long heralded and is well understood.

* Within Western Europe, the particular role of the PTTs and the vast financial resources required to build a major network capability determine the emerging structure of this new sector.

* Fundamentally, this can be viewed at three levels.

* The monopoly or duopoly of basic telecommunications bearer services that exist in Europe determines the top level.

* At the next level, the possibility exists of developing major service network capability. Clearly, a number of service companies, notably GEISCO and McDonnell Douglas, already offer comprehensive services of this kind.

* Increasing technical capabilities as a result of convergence are making large companies like GE, IBM and EDS offer managed data network services or VANS.

* At the third level, a highly fragmented sector is anticipated in which many organisations, several small and highly specialised, will offer particular services (VANS).

* These organisations are likely to purchase network capacity from second-level vendors and will generate value-added on the basis of the data or related information services that they can provide to their clients.

* Examples of these services include **electronic data interchange (EDI)**, on-line data base services, and **electronic funds transfer (EFT)**.

Software products market growth

* INPUT forecasts a gradual maturation of the software products market and a slowdown in growth rates in the period up to 1991.

* The four major European country markets are expected to grow at an annual rate of 26 per cent for the period through 1988 to reach $7.3 billion. Growth is expected to fall to 23 per cent during the period 1988 through 1991, with the market size expected to reach nearly $14 billion (see Fig. 9).

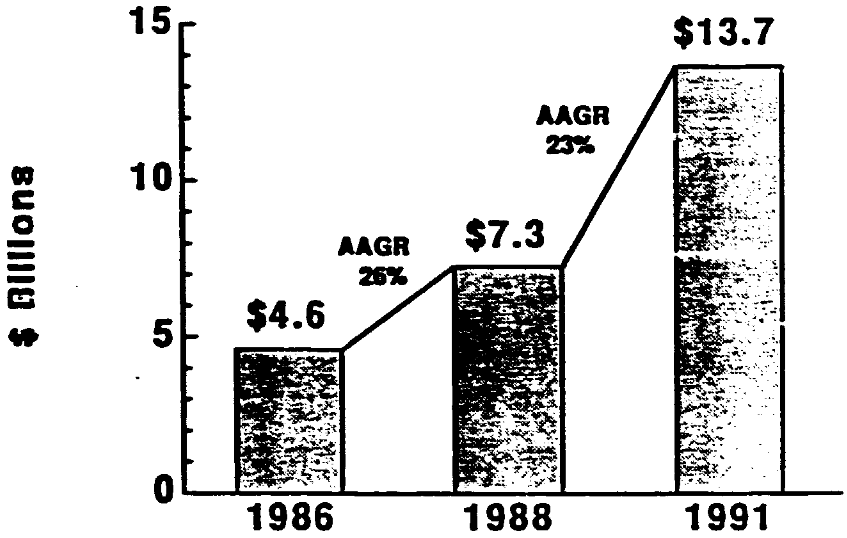
* One of the major threats to the growth of the software products market is the decline in prices for low-end products resulting largely from highly competitive "commodity-like" characteristics in the personal computer (PC) area.

* Consideration also has to be given to the capability of organisations and individuals to continuously absorb new products.

* Nevertheless, marketing clearly superior value-added software products in key application areas will continue to offer major opportunities.

* Major customer bases, hitherto not great absorbers of software application packages (e.g., the VAX BASE), are likely to represent a sizeable opportunity in this five-year time frame.

Fig.9: Software products market growth
(average annual growth rate)



<u>**Software vendor outlook**</u>

* The marketplace for software products remains fast-changing and provides rapidly growing opportunities.

* Key characteristics are the shortening life cycles for products, the competitive turmoil, the ever-present threat in the complex systems area from the professional services approach, and the unforeseen activities of the equipment manufacturers.

* Important vendor concerns must be to constantly improve or replace products in order to remain "state-of-the-art" and, perhaps more importantly, keep abreast of rapid changes in market demand.

* As "low-end" product prices continue to fall, greater emphasis will need to be placed on unique (and valuable) features to products that will be able to justify higher prices in order to cover heavy development and marketing costs.

* Major new opportunities over the next five years are seen in areas like the application of artificial intelligence technology to software products in general.

* Exciting opportunities are also seen in the area of productivity tools. The application of artificial intelligence technology to this area will be significant.

* Increasing specialisation is also observed in the software products market. Vendors will need to provide products targetted at specialist markets and to demonstrate their knowledge of the relevant business areas when marketing and supporting the products.

Professional services market growth

* INPUT estimates the growth rate of the **professional services sector** at around 25 per cent per year, leading to an annual market of $9 billion in the four major countries of Western Europe by 1988 (see Fig. 10).

* In the longer term, uncertainties surrounding the availability of skilled staff and the speed with which productivity development tools can be utilised are likely to represent supply-side constraints.

* On the demand side, there are many development opportunities as users increasingly seek total service solutions.

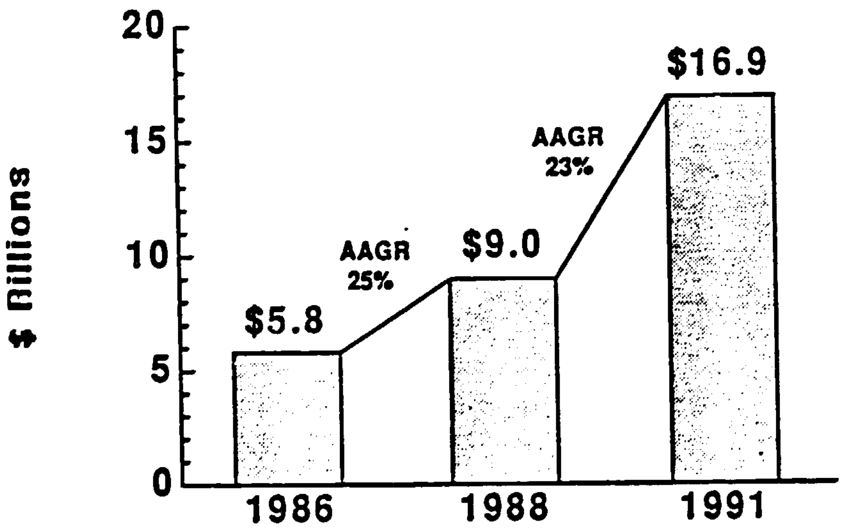
* Education and training has emerged as major opportunities due to:

  - rapid changes in technology;
  - growing complexity of software development tools; and,
  - increased penetration of microcomputers among inexperienced end-users.

* The provision of consultancy services is growing in importance as users seek to implement company-wide systems strategies.

* Specialisation is the key to success in systems implementation owing to the proliferation of increasingly complex technology and the need to demonstrate specific knowledge of target sectors.

Fig. 10: Professional services market growth (average annual growth rate)



Professional services trends

* The trend towards the use of information systems as a key strategic weapon is increasing the demand for professional services.

* The increased demand for large and complex systems, particularly those dependent on telecommunications networks, leads to the need for contractors to take on overall responsibility for systems integration projects.

* A key trend here is the move towards **prime contracting** and the emergence of **systems integrators**.

* Factors stimulating the systems integration marketplace include:

  - scarcity of skills by one vendor for developing a total complex automated solution; and,
  - proliferation of technological options, producing buyer challenges and system compatibility changes.

* Consultancy services are being utilised in areas where rapid technological development leads to a scarcity of in-house skills. Telecommunications and expert systems are good examples of this.

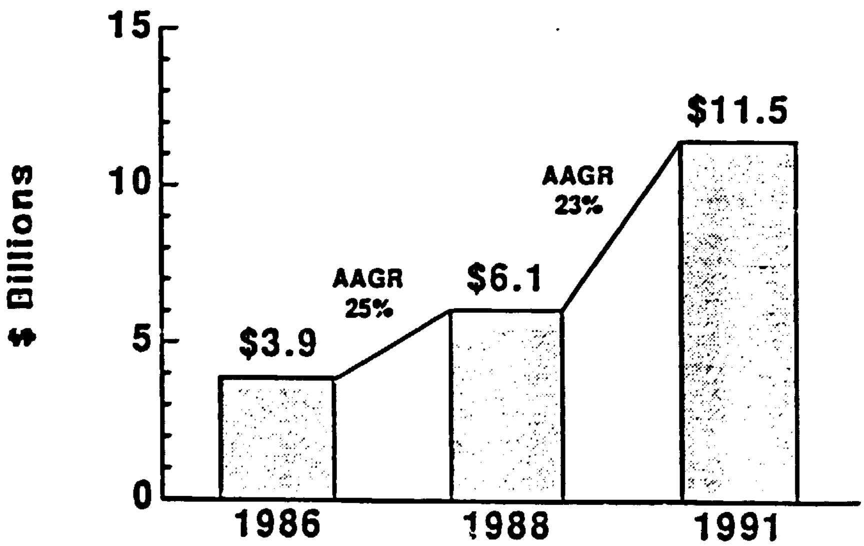
* The "Big Eight" accounting firms have increased the market for professional services and have enhanced levels of professionalism in the sector. However, there is still a cloud over the ethical issue of impartiality in their auditing activities.

<u>**Integrated systems market growth**</u>

* The integrated systems market encompasses three main areas: CAD/CAM, small- to medium-sized business systems, and specialist vertical or niche market systems. Vendors typically provide a combination of hardware, software and services.

* Currently, Europe is experiencing a relatively high rate of growth (25 per cent average annual growth rate). INPUT forecasts that the market will reach $11.5 billion in 1991, having reached $6.1 billion by 1988 from its level of $3.9 billion in 1986 (see Fig. 11).

Fig. 11: Integrated systems market growth (average annual growth rate)



* The major forces driving this growth are:

  - Increased awareness of the use of automation as a major competitive weapon. This attitude is fuelling demand for rapid implementation, which favours "ready-to-go" integrated systems solutions.

  - The increasing appeal of a "one-stop" service, relieving confused buyers of a time-consuming, complex and error-prone approach to hardware/software evaluation and selection.

  - The continuously improving price/performance of both mini- and microcomputers, opening up new markets by lowering the entry threshold of integrated system solutions.

  - The healthy growth of the software products marketplace. Integrated systems vendors will have a constantly expanding menu of innovative software solutions available for bundling into a total integrated system.

<u>Integrated systems challenges</u>

* Increasing specialisation of user needs has allowed vendors to leverage expertise in specific market segments. It is important that vendors provide all the value-added service factors for specialist markets.

* The provision of implementation, consultancy and support is a key success factor. Specifically, it is suggested that vendors consider undertaking more extensive pre-installation systems

consulting, education of users in the managemen` principles underlying the architecture of the application, and innovative hardware/software problem reconciliation methods (e.g., remote diagnostics and on-line access to fault diagnostics data bases).

* Sophisticated software engineering is a major challenge, and it is suggested that vendors focus on multi-user applications and the customisation of standard modules of software in order to enhance development productivity and maximise market penetration.

* The CAD/CAM market has undergone a mild "shake-out" phase as vendors have adjusted to slowing growth rates, falling unit prices, and increasing implementation of CAD systems on standard hardware (especially microcomputers) rather than expensive proprietary systems. It is suggested that vendors concentrate on growing opportunities in software publishing and the provision of value-added via comprehensive support, consultancy, and training services.

Conclusions

The ratio of hardware to software costs has now reached 50:50. Software costs will still increase rapidly and hardware costs will fall constantly. The application software market is growing substantially in specific key application areas. This will offer a mature opportunity to software houses working in these areas. The available application software is unstructured and polluted, requiring high maintenance costs. The application software has therefore to be sanitised in the next 15 to 20 years. The only chance of improving the software quality is to build highly-sophisticated software development environments.

All of this implies an increasing growth in the areas of software products and professional services. Their average annual growth rate will reach 24 per cent in 1987 to 1991. The integrated systems sector is expected to grow at a slightly lower annual growth rate of 23 per cent. The processing and network services sector cannot hold its share of 25 per cent in 1985 and will fall to a 13 per cent share of the market in 1991.

The emphasis on introducing software products instead of individual software development will lead to a continuous gap in education and training. This area has to be developed much more professionally.

III. THE STATE OF SYSTEMS DEVELOPMENT AND SYSTEMS APPLICATION IN DEVELOPING COUNTRIES

Software pollution: lessons for developing countries

The state-of-the-art in data processing in industrialised countries is not worth imitating. Through rationalisation of hardware production, the hardware/software cost-ratio is now 20:80 and is predicted to be 10:90 in 1988. Programming staff in industry and in public administration devote 80-90 per cent of their total manpower to software maintenance. The value of installed application software worldwide is about $ 150 billion to $ 200 billion. Nearly all these software systems are badly structured and difficult to maintain. This is called "polluted" software.

To reduce maintenance costs and to ease adaption to market, technology and legal changes, the polluted software has to be restructured and partly rewritten.

The reasons why software pollution takes place have not always been understood from the beginning of application software production. Now, it is known from empirical studies as well as from theory that there are normal thresholds which cannot be overcome. Indications are that the best possible software that can be created has about 0.5 "bugs" per 1,000 lines of code (LOC) on average. This means that existing operating systems, compilers and data base management systems contain about 150 to 300 "bugs" and even more, because of their size of 300,000 to 600,000 statements. This observation has also been confirmed by theory. The complexity theory, a special subject area of computer science, shows that even for very trivial problems, the number of steps in algorithms grows super-exponentially.

For example, the number of steps to decide whether a given string of natural numbers, brackets and arithmetic operators fulfills the axioms of the so-called Pressburger Arithmetic (a trivial arithmetic for adding natural numbers), grows with (2^2) to the power of n if n is the number of characters of the string. This growth is faster than the exponential function, which is well known for enormous growth.

Experience in software production of complex systems in industrialised countries as well as results of complexity theory show the danger of following this path. Developing countries still have the chance not to repeat the same failures and not to run into the same problems as the industrialised countries. It is worth emphasising once more that a governmental strategy and policy is urgently needed to avoid a situation of having totally polluted software. Governments of developing countries have to react fast because microelectronics and the related software have already started penetrating these countries.

What solutions could help cope with this situation? One problem is the concentration of too many functions in one system and the human inability to structure and put layers in a homogeneous manner into these systems. The solution would be to decompose systems, also artificially if necessary, in different components running on different computers in so-called distributed or loosely coupled systems.

A second solution is to involve the user for whom the system is being produced in the problem-solving process from the beginning. How should this be done efficiently and effectively? The systems designer and programmer must have software tools capable of producing a prototype system quickly. The designer can give the user, with this rapid prototyping strategy, a preview of the future system after having spent only 15 to 20 per cent of the whole project costs. Today, the acceptance test of the user is done after total implementation of the system and its testing. With rapid prototyping, the first acceptance test is done after 15 to 20 per cent of the work is finished. In the past, the acceptance test showed far too late the failures and misunderstandings originating from the design process.

A third solution is to define a skeleton of interfaces between different software components. The basis of this solution is the following: If we do not restrict ourselves to snapshots of our hardware and software components but instead consider our data processing environment over a period of 20 to 30 years, we will recognise that we cannot rely on hardware, on operating systems, and on higher programming languages or data base management systems. We can rely only on so-called virtual interfaces, building a skeleton of interfaces as for example the so-called virtual terminal interfaces. The functions of the virtual terminal are specified without regarding a specific terminal. Keyboard functions, function codes, characters, digits and special characters are specified without allocating special signals. Interfaces adaption from the virtual interface to the existing physical interfaces is done by software. The skeleton of interfaces allows easy change of equipment and software components.

Developing country constraints

Observations on the data processing market in developing countries are listed, showing the complex interrelationships and the psychological and economic constraints. To some extent, they are also valid for industrialised countries.

- There are only a few software companies or other so-called third party (consulting) companies working successfully in the market.
- Hardware manufacturers and distributors are acting mainly in the data processing market.
- Hardware is oversold and there is little appreciation of software and systems (=hardware + software) thinking.
- Complete solutions tend to be imported by companies through hardware manufacturers and distributors in standard software packages.
- Normally, standard software packages do not fulfill user requirements totally.

- Consequently, the organisation has to adapt to the data processing instead of vice-versa.
- The tendency is for highly-educated staff to leave the industrial and university fields and join hardware sellers; or they go to the industrialised countries.
- There is no confidence in establishing third party companies (software houses, consultancy companies) because of the belief that software development is too difficult.
- Nobody admits troubles being experienced with the "status symbol" computer.

Possible entry points for developing countries

The worldwide microelectronics market shows that from the economic point of view, only parts that build up the basis for an independent computer and software market should be produced in developing countries. The design of systems and the assembly of hardware and software components into sytems is feasible and challenging. Educational standards in developing countries are adequate. Psychological, financial and managerial barriers have up to now hindered the development of an indigeneous data processing market in developing countries.

The state of the microcomputer market reveals that microcomputer application systems can be introduced in developing countries inexpensively. These systems can support in a distributed manner industry, commerce, public administration, banking and insurance companies in the management of masses of data (text processing, data base systems, information systems, statistical computation), control of processes (manufacturing, quality control, optimisation procedures) and government and industry planning (model building, simulation, optimisation). Standardised microcomputer operating systems and programming languages will ease the penetration of microcomputer application systems into the market.

The software market shows a constant increase in software costs against a constant decrease in hardware costs. "Software costs" cover the design phase through the installation phase to maintenance; software development is a continuing process throughout the life cycle of the software. The full software life cycle and software maintenance today cover about 70 per cent of total costs. This makes today's ratio of software to hardware costs in the industrialised countries 80:20 and 90:10 in 1988. The trend is, more and more, to cost software and hardware independently, resulting in a so-called unbundled software market. This opens opportunities to developing countries since they do not have a lot of polluted software yet.

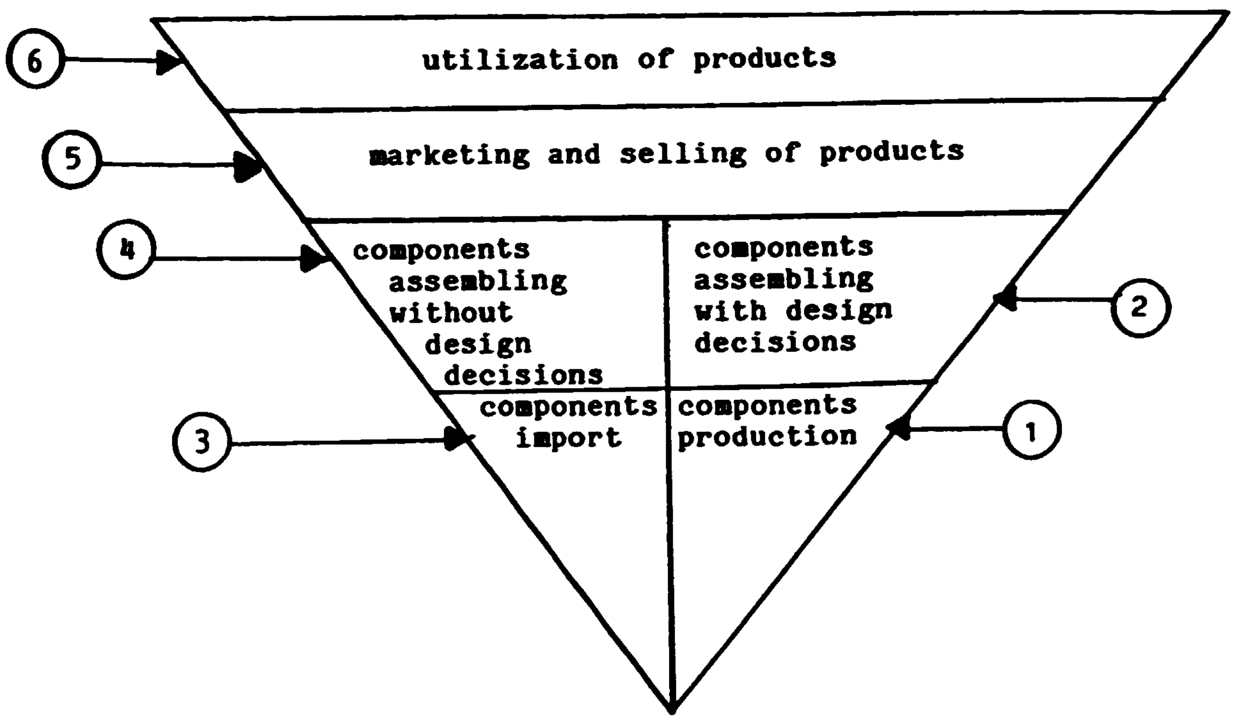
Several countries have to import software technology know-how and to export software services. Examples abound: From Hungary to the Federal Republic of Germany, from Singapore to the U.S.A., from Poland to Austria, from India and Taiwan to Great Britain.

A big market of software tool boxes for so-called software development environments is also opening for microcomputer applications. In contrast to the hardware industry, which is dominated by a small number of huge multinational companies, the software industry consists of many small- and medium-sized enterprises.

In developing countries, the emphasis is on hardware aspects. The experience in industrialised countries has also shown that systems as an integration of hardware, basic software (compilers, utility programs) and application software are the import units considered. Software aspects are fully underestimated. There is no consciousness for software development and application software tools.

In Figure 12, different entries are marked describing the degree of technical and economic development of a country. Some developing countries that are much closer to the industrialised countries than others have the tendency to start with entry 3 or 4. To start with entry 1 and 2 would increase international competitiveness.

Fig.12: Entry points of technological development



IV. POTENTIAL OF NORTH-SOUTH CO-OPERATION IN SOFTWARE DEVELOPMENT IN DEVELOPING COUNTRIES

The need for computer software is growing enormously and the market for software products is rather open. Software production is extremly labour intensive and developing countries can have a substantial advantage in producing software at lower cost relative to developed countries. Thus, software production in developing countries may be highly competitive in the near future. In addition, special applications in developing countries can be promoted effectively only with a good software capability.

Industrialised countries have spent about $ 150 billion to $ 200 billion worldwide for systems and application software development. Hardware costs are still decreasing by a factor of 10 in 10 years; costs for highly skilled personnel are increasing. The only way out of the so-called software crisis is to standardize software and support software development by software tools. Industrialised countries have to reorganise and to reconstruct their polluted software in the next two decades. Developing countries now starting software production have the chance to avoid all the past mistakes of the industrialised countries. They could use software tools for software production just as machine tools were used in the industrialisation process. One way is to define a so-called skeleton of software interfaces in which all software building blocks developed within a time period of 5-10 years fit into. IBM,with its System Application Architecture (SAA), is now offering such a strategic skeleton of interfaces.

Another big problem with software and microelectronic products is lack of reliability after deli ry. While more sophisticated purchasers are interested in the quantifiable dimensions of quality and reliability, in markets for software and consumer products perceptions - whether or not well founded - influence the decisions of prospective customers.

If there are trained systems analysts and software engineers available, it is possible to reduce maintenance problems. The specialists, knowledgeable about computer systems and their problems, improve the reliability and quality of products. This competence will help create and strengthen indigenous core groups of professionals in developing countries capable of undertaking software and product maintenance. Software maintenance involves either correcting errors that went undetected in development, or changing programs as a result of altered or additional requirement specifications.

In the past the area of systems integration was very important. A systems integrator combines standard hardware components with customized software - or certain standard software packages modified appropriately - in a unique configuration more or less tailor-made to end users' requirements. There has been a trend since 1979 towards the production of standardized packaged software and away from customized production and integrated systems software. For example, between 1981 and 1983, packaged U.S. software grew at a 40 per cent annual rate compared to increases of 22 per cent for integrated systems software and 16 per cent for customized production. Software houses are under pressure to produce larger, more complex, more reliable, and more cost-effective systems and these pressures are creating market niches for small firms, niches too narrow and specialised to attract large firms.

Opportunities for developing countries in exporting computer software lie in the areas of software customization and modification, systems integration, and in related support services. Opportunities appear greatest in the innovative development of applications for specific industries and in the customization and modification of cross-industry applications. Entering vertical markets will require, among other things, identification of selected niches left unfilled by the larger firms to be targeted for software design.

Thus, a firm may more easily fill niche market demand for specialised software products. Further, the needs of computer users are generally specific to a particular country, organisation, and environment. Because laws and regulations of developing countries for payroll deductions, income tax withholdings, social security, workmen's compensation and so forth differ from those of developed countries, production of local versions of software is critically needed. Because the number of specialised users of information technology is growing worldwide, the relative importance of software firms able to respond to the specialised needs of small segments of the user population will likely increase.

Developing countries could begin with supporting software groups in governmental, university and industry environments. These groups could start with improving user interfaces of existing installations. As a follow-up, software tool boxes from developed countries should be imported to produce well structured (and not polluted) application software.

How to start activities in different types of developing countries? First of all, the government has to have an overall strategy for micro-electronics and software production. According to the level of technological development, the technological requirements are different. If application software is the target for development, hardware maintenance and some system software support is sufficient. International contacts with software companies in industrialised countries support technology transfer of application software development tools. After having stabilized software production in a country, South-South and South-North exporting of software and services could be tackled. Bodyleasing from South to North should not be supported. In less advanced developing countries, emphasis should be laid upon curricula development and education at each level of sophistication.

The Singapore model

The objectives of Singapore's governmental policy are described, revealing an outstanding approach towards reaching competence in information technology.

The economic restructuring plans of Singapore include the country's development into a software center for the region. To be successful, a software center should have significant applied research elements in the software projects undertaken by computer companies.

The new phase of Singapore's technology drive have the following national objectives:

(a) Creation of an appropriate environment in which technological innovation is encouraged and rewarded.

(b) Development of an applied research capability which will influence technological developments in Singapore, directed at increasing the productivity and competitiveness of Singapore organisations, and supporting a successful export-oriented software industry.

(c) Acceleration of transfer of new and emerging technologies to Singapore.

Among the responses to these national aspirations is a new Research Division that has been set up at the Institute of System Sciences (ISS). The mission of this division is to establish a premier applied research group and to disseminate results from an on-going program of applied research, which will help in the formation of a strong technological base for a successful computer services industry in Singapore. The objectives of the Research Division are to:

(a) enhance the environment for applied research;

(b) help develop the necessary skills base; and,

(c) enable the transfer of R & D knowledge, tools and techniques to Singapore's enterprises.

ISS will be an important conduit for transfer of technology. A number of visiting experts will be invited. Prominent academics and researchers will be encouraged to spend their sabbatical at the institute. Assisting in the selection and appraisal of research projects will be an International Advisory Board.

An important thread running through the institute's work will be strong links with industry. Joint projects with industry will be encouraged and representatives of industry will be welcome to spend time at ISS. To multiply the impact of project work at ISS, staff members of academic departments at the National University of Singapore will be involved. Two organisations that will work closely with ISS in joint research projects are the National Computer Board (NCB) and the Telecommunications Authority of Singapore (Telecoms). NCB is increasingly giving more attention to the establishment of a strong technological base with applied R & D capabilities. A Joint Software Engineering Program, established in 1983, pools computer and software engineers from NCB and the Ministry of Defence. Work is being carried out on software analyst workstations, integrated office systems and expert systems.

Singapore is considered as having one of the most advanced telcommunications facilities in the region. Telecoms is carrying out research on several projects including Teleview, Telemetering, Optical Fiber Networks and Cable Video Systems. The Teleview project will be a massive program, designed to be an interactive system for disseminating and retrieving computer-based information using public telephone lines for communication and the home television set for display.

The Research Division would undertake advanced information technology research. The objective would be to play a leading role in elevating the technological level of Singapore's information technology industry and to be a training centre for research personnel. One project being explored is a public information expert system with the data and video recording stored in an optical disk. Under a partnership program with IBM, the institute will be able to tap into the technology of the computer company and expose ISS researchers to state-of-the-art research. ISS researchers may be posted to various IBM research laboratories for internship assignments of up to one year. Similarly, the institute will accept interns from Singaporean organisations interested in information technology research.

The Education Division is responsible for all teaching activities. Its staff may look forward to a highly rewarding teaching role. A course development model has been put in place to ensure high quality presentations, visual aids, student hand-outs, and documentation. The model requires considerable interactive group effort in course development to provide solid linkages from topic to topic. Continuity of courses will take into account recently introduced technologies and methodologies.

Staff members will be working in groups, resulting in considerable sharing of experiences. The ISS teaching staff are professionals with many years of computer experience in industry. Highly sought after for advice and guidance on a number of information systems projects in Singapore, their expertise ranges across a number of key areas including management of information systems, data base management systems, data communications and software development. Their expertise also covers a number of different brands of computer equipment and software products.

The teaching staff have the opportunity to mold some very bright postgraduate students into highly skilled systems analysts who are likely to be the future project leaders in key Singaporean organisations. Through the short courses, ISS staff members are in regular contact with a host of managers and computer professionals. The staff member plays the role of educator, teaching the students concepts, techniques and technology; of advisor, tracing through the intricacies of technology and the process, value and pitfalls of implementing technology; and of facilitator, managing group discussion and encouraging critical thought. Staff members have the opportunity to influence key organisations in Singapore to move along innovative technology paths in an effective manner.

To ensure that technological development in Singapore remains dynamic, it is important that ISS staff members are creative, progressive and keep themselves up-to-date in their areas of expertise. ISS will provide every opportunity to bring in visiting experts to brainstorm with the staff and to conduct seminars. Staff will also have the opportunity to represent ISS on technology watch groups, advisory committees and technology planning groups, some of which have national missions. This ensures that staff keep in constant touch with industry needs. ISS staff may also be sent abroad on conferences and internships. ISS staff members are encouraged to read widely and carry out small experimental projects. Some of the projects currently under way at ISS are:

(a) Office Systems Development
An experimental network links up all staff members. Studies on network performance, document processing and gateways are being done.

(b) Applications Development Languages
Work has been done to survey fourth-generation languages and data dictionaries; effective implementation of these tools will be explored. A parallel effort is going on in understanding software engineering and in utilising languages such as ADA for applications development.

(c) Information Systems Planning

ISS is studying several methodologies and has assisted several Singaporean organisations in carrying out strategic information system planning. A concerted effort is under way to study and explore the links between enterprise analysis, business systems planning and data analysis. Consultancies have been undertaken on a limited basis with major organisations, ensuring that ISS members employ and sharpen their technical and management skills, upgrade their expertise, and have a better understanding of organisational needs in Singapore. Organisations have found these consultancies to be very effective in accelerating their information systems planning. Consultancies are ISS-sponsored, with contributions by ISS management and relevant staff members to provide and ensure high quality service to its clients.

V. NEW TECHNIQUES OF SOFTWARE PRODUCTION

Computer-assisted problem conception technology (COMPACT)

Today, software development can be planned only on a range of 7 to 10 years: This implies a long-range concept of the system development process and of the production supporting tools. Computer-assisted problem conception technology (COMPACT) is such a methodology concept. On two different levels of abstraction, Fig. 13 and Fig. 14 emphasise some classic tools for the design and realisation phases. (For further information, see /Bu80/, /Me79a/, /Met78/, /Nu75/, /Scn79/, /Scn82a/.)

Fig. 13: Overall system architecture of the COMPACT methodology

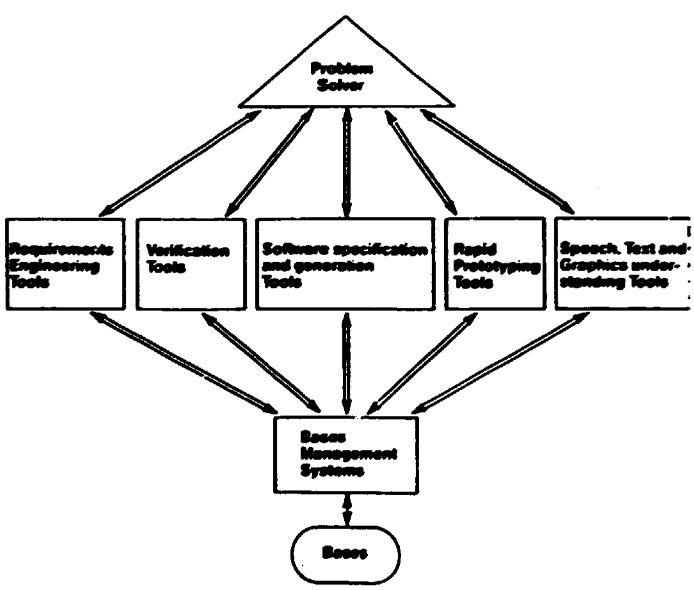
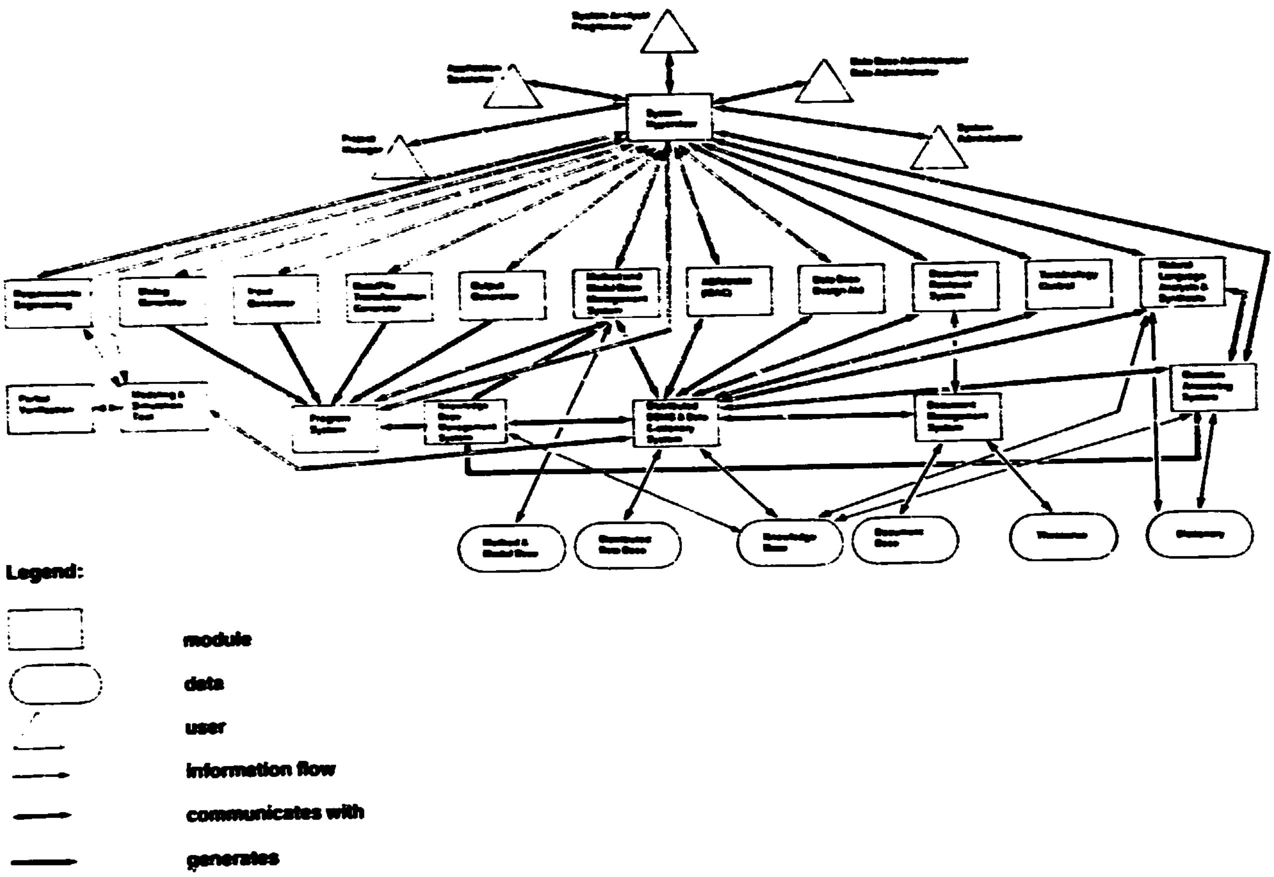


Fig. 14: Detailed system architecture of COMPACT



Some of the tools of COMPACT (Fig. 14) follow.

- The **ISAC component** consists of a systems design and specification technique. ISAC has interfaces to all phases of systems development, starting with problem analysis up to implementation, test and documentation of the system.

- The **terminology control component** takes care of the control and handling of an enterprise's or administration's terminology, maintenance of thesauri, and the semi-automatic indexing of texts in a uniform and consistent way.

- In general office work, the handling (maintenance and retrieval) of documents has become more and more important. Therefore, a **document information system** that has all the relevant features of **document retrieval systems** will be integrated. Combined with the **terminology control component**, the system is a helpful tool for content handling within the application environment. Control flow and data flow pseudo-languages are described in /Le79/.

- In operations like data input, data output and data deletion, the automatic generation of input and output forms is considered very important. The **input and output constructors** are modules that establish programs for the automatic generation of problem-oriented input and output. This is assisted by an application-oriented screen mask specification of tasks by means of an interactive terminal dialog for different problem classes. Such systems are described in /Ca81/, /En80/, /He80/.

- The **automatic data and file transformation component** reduces the mass of bridge programs needed every time transformation of input and output formats of programs is necessary.

- Today's industrial and administrative applications have made the development of dialog-oriented user interfaces even more important. Therefore, the **dialog constructor component**, for the specification and automatic generation of any desired dialog program, is needed (/Stu80/,/Was81/).

- The integration of the **method and model base system** component (also known as **decision support systems**) may prove useful in a further stage of development. Some valuable examples of research efforts within this area are described in /Be81/, /Kon81/, /Lo81/, /Me81/, /No81/.

- The whole concept is built upon a **data base management system** (centralised or distributed) with a powerful **data dictionary** and, if available, with a **database design aid**.

Main utilisations of complex tool systems like COMPACT will be in office and production planning and automation. An example of future systems, CIM-software development, is described.

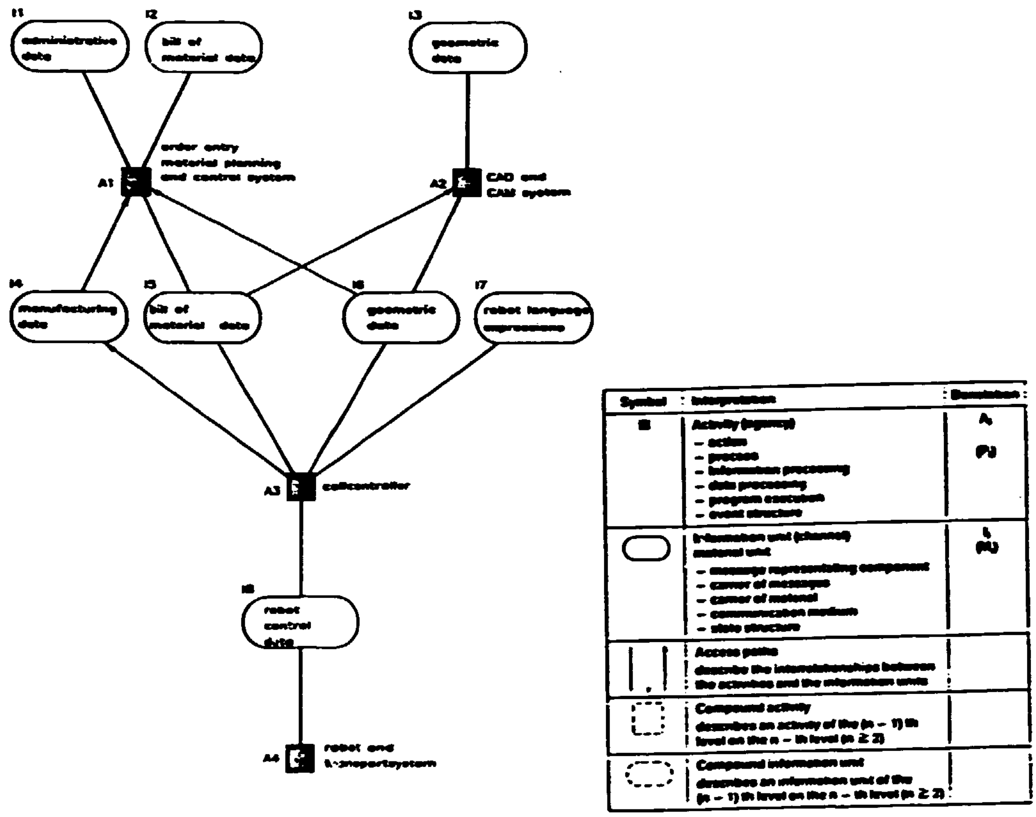
Computer-integrated manufacturing system (CIM)

Installation of new knowledge-based manufacturing systems - i.e., systems of the fifth generation - will lead to significant increases in industrial productivity and flexibility through late customising of products. Developments in most application areas of industrial production will result in computer-integrated manufacturing.

What does a typical CIM solution look like? First, a new part is defined geometrically with an interactive graphics program for computer-aided design (CAD) and manufacturing (CAM) running on a host system. The corresponding data needed for production planning will be collected in the bill-of-material system (BOM). Data will be manipulated by a system for order entry, material planning and control running, also on the host computer system.

Robots controlled by robot-language expressions and the geometric and bill-of-material data downloaded from the host system to a cell controller and robot controller then carry out (for instance) the mechanical assembly or the component insertion. Additional information about joining points of the various manufacturing components (e.g., transfer stations, transport system, bowl feeders, magazines) and sensor data support this process. After the order is completed with administrative data, all manufacturing data (numbers of good and faulty parts, consumption of subparts, execution time needed) are sent back from the cell controller to the production planning system in order to initiate delivery notes and invoices printing. The cell controller, typically a personal computer, is used to create, compile, and download the robot-language expressions to the robot controller. Fig. 15 shows the data flow and components in such a CIM-system.

Fig. 15: Data flow in a CIM-system



Knowledge-based manufacturing systems of the fifth generation will allow an increase of profit and revenue in the face of further sharpening competition and rapidly changing technologies. These systems have to provide a link between the four major islands of automation: engineering design, production planning, manufacturing control and factory automation including material handling. Figs. 16, 17 and 18 show a functional and data flow diagram of the CIM-process. Artificial intelligence concepts and expert systems technology will support the planning, installation and running of such complex systems.

Fig. 16: Functional diagram of the manufacturing enterprise

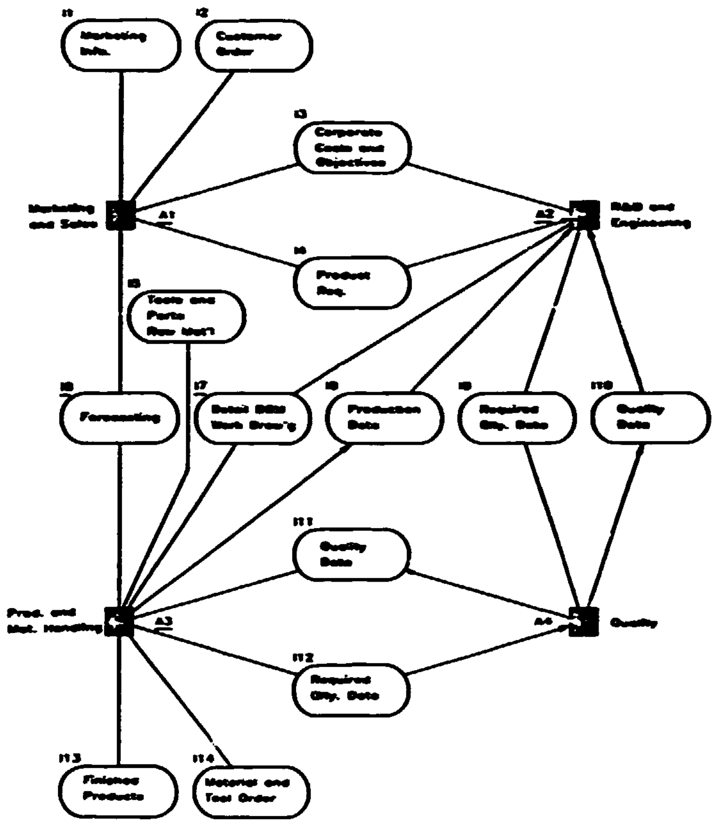


Fig. 17: Refinement of agency "A1: Marketing and Sales"

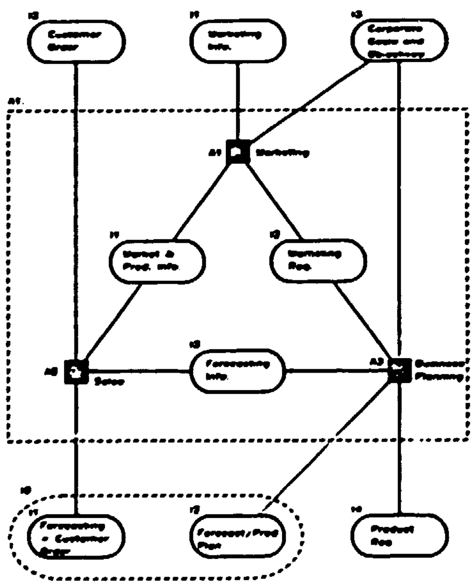
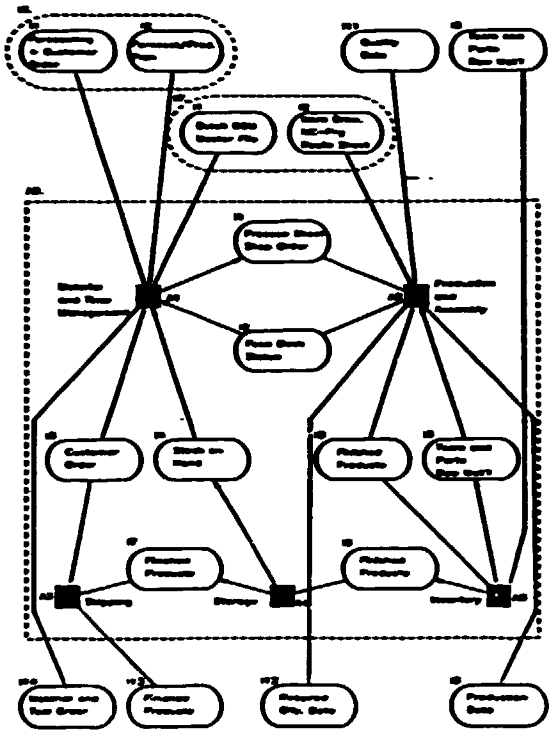


Fig. 18: Refinement of agency "A3: Production and Material Handling"



Up to now, CIM has been meant as a strategy. A CIM software product is not yet available. There are several theories as well as practical ways on introducing a CIM strategy in a company. The main approaches known are Manufacturing Resource Planning (MRP II), KANBAN of Toyota Motor Company, and Just-In-Time (JIT) production. Further approaches have to be developed because of the disadvantages of the existing ones.

Knowledge-based systems of the fifth generation

Up to now, single technologies such as microelectronics, software technology, fiber-electronics, and telecommunications have been the focus. In the future, computer technology, integrating methods, tools and strategies to develop systems will be emphasised.

The Japanese program for fifth-generation computer systems has had an unexpected worldwide echo effect. For the first time, combined government and industry R & D activities of one country has triggered substantial follow-up research activities in other countries. With this program, Japan aims to be number one in the world in information technology. Japan also initiated, in 1981, a new institution related to complex research, the Institution for New Generation Computer Technology (ICOT). More than 50 researchers were fully employed in 1986 in this Institute. ICOT grants contracts to industry partners.

Japan's fifth-generation program triggered these responses from other countries:

- The DARPA program on Strategic Computing and Survivability was submitted to the U.S. Congress in November 1983.

- The U.S. National Science Foundation announced a new initiative for the development of super computers in November 1983.

- The U.S. Department of Defense announced a long-range plan for systems of the "Nth Generation" in 1983.

- Fifteen well-known American companies have initiated a new company, the Microelectronics and Computer Technology (MCC). It will have an annual budget of $ 50 million to $ 100 million and is meant as an answer to the Japanese challenge.

- The European Commission has started the ESPRIT Program (European Strategic Program for Research and Development in Information Technologies). The first five-year plan covers a budget of DM 3-4 billion for R & D.

- The British Government has started the ALVEY Program, similar to the ESPRIT program.

One of the main goals of the Japanese program is to supplement classic data processing with knowledge processing. This goal requires deep insight in human processes like problem description, problem-understanding and problem-solving, and closely related subject areas like knowledge representation, knowledge processing, knowledge acquisition, dialog handling, and the analysis and synthesis of natural language. New formal calculi, artificial languages, and methods have to be developed to solve all these problems. In the past few years, special logical theories, new logical programming languages (PROLOG), knowledge representation and processing languages, and system architectures have been developed.

The goals of the Japanese program are ambitious. If they are reached, a series of milestones will have been overcome:

- The application specialist will develop his own application programs.

- Maintenance of manually specified and (semi-) automatically generated software products will be much easier because of the unique programming style and the standardised interfaces.

- The problem-solving quality (the highest goal possible) will be improved drastically because users can concentrate on the solution process and will not be distracted by data processing details.

- Software development will be an engineering discipline (traditionally-tailored software development will become obsolete).

- Systems will be able to recognise failures, inconsistencies, discrepancies and contradictions; in most cases the systems themselves will be able to correct and solve the mistakes automatically.

- Systems will contain powerful, fast, easy and reliable tools for knowledge acquisition (i.e., knowledge transfer from the expert to the system).

- Systems will offer a substantial increase in programming productivity.

The Japanese researchers see the following as the main application areas of fifth-generation computer systems:

- automatic translation systems of natural language texts with a vocabulary of more than 100,000 words and an accuracy of translation of about 90 per cent;

- expert systems in various application areas with 5,000 different words and 10,000 inference rules;

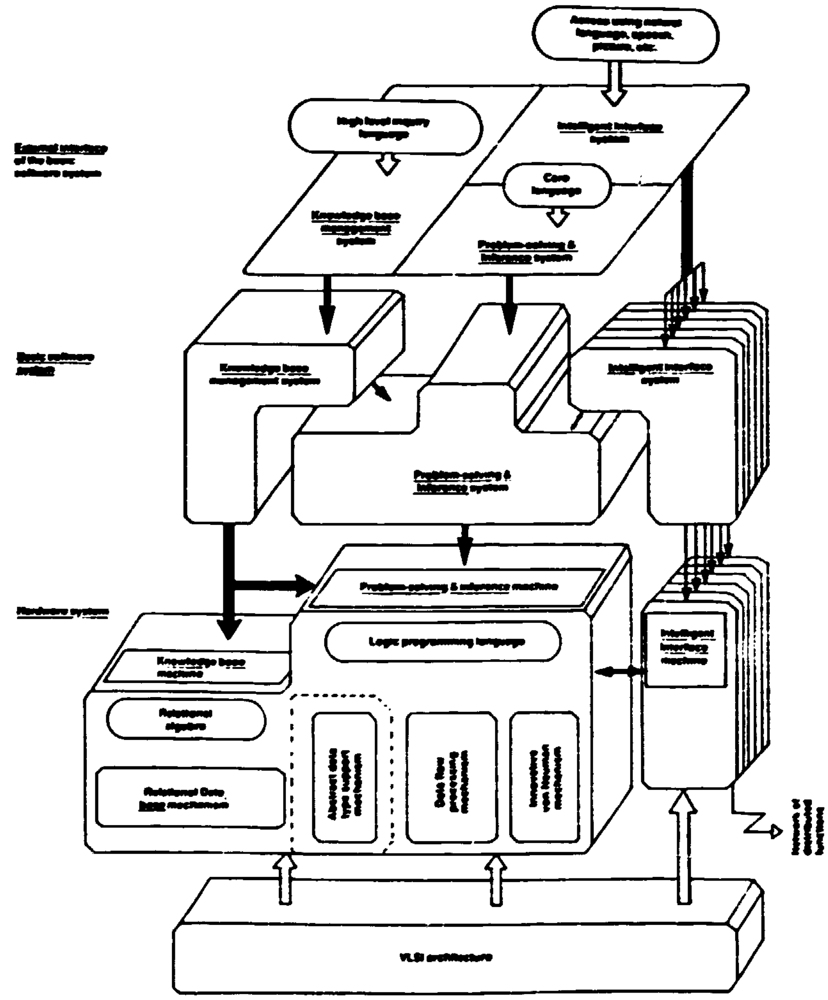
- knowledge-based software development environments;

- knowledge-based VLSI (very large scale integration) chip production and CAD, CAM, CAE, CAP (computer-aided planning) systems, and robot systems;

- knowledge-based problem-solving systems for specific application areas, e.g., in office automation, automatic dictating machines with a vocabulary of 10,000 different words and speaker identification covering a sample of 500 different speakers. In addition, there will be important applications in medical diagnosis, in juridical and tax query systems, in chemical structure analysis, in detecting new oil and mineral sources, and in failure-detection within systems.

Fig. 19 illustrates the original architecture of the fifth-generation computer systems developed by Moto-Oka and his research colleagues. The hardware architecture of these computer systems is no longer von Neumann architecture. The software field is also heading towards unconventional directions.

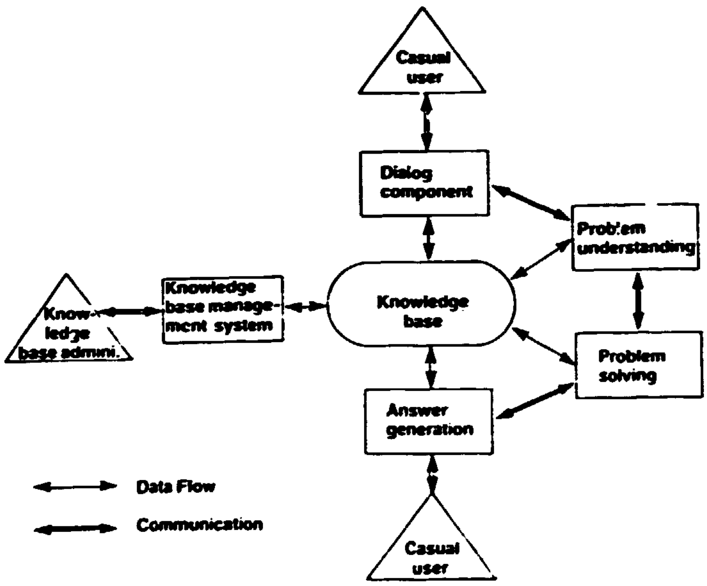
Fig. 19 : Architecture of fifth-generation computer systems (FGCS)



The requirements of FGCS for hardware speed lie between 10^{12} and 10^{14} instructions per second. The data base machines should be capable of storing 100 to 1,000 billion bytes, the intelligent interface language, 2,000 grammar rules, and about 10,000 different graphic elements.

Fig. 20 illustrates the main components of FGCS architecture. The figure shows the most important software research and development areas: **problem solving, knowledge-based processing,** and **intelligent interfaces**.

Fig. 20: Data flow diagram of a knowledge-based system



Hardware specialists will build the first layer in VLSI technology; software specialists will develop the corresponding software components and interfaces for the casual user.

Fig. 20 shows a data flow diagram of a knowledge-based system for general problem-solving:

- The **dialog component** needs concepts and techniques from linguistics, logic, artificial intelligence, and cognitive science to be able to map natural and quasi-natural language expressions into formal representation language expressions and vice-versa. The casual user will have a very user-oriented natural interface.

- The **problem understanding component** analyses the content of the language expressions. It must represent and process spatial and temporal knowledge, differentiate between facts, events, and actions, and construct a model of the real world, its objects and dialog partners. This component must recognise whether a given string is, for instance, a proposition, a query or an instruction. Corresponding to this recognition process, a command procedure for data base input and/or change, query or program generation must be created/generated. It must evaluate the transferred knowledge, whether it is vague, reliable, evident or relevant knowledge. The component must recognise failures and contradictions and must protect confidential and secret information.

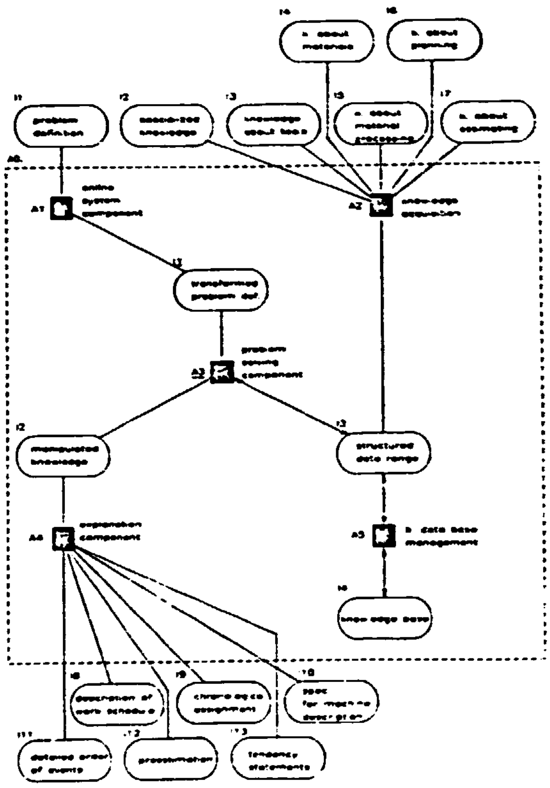
- The **problem-solving component** then processes, e.g., data base transactions, translation to lower levels of abstraction, program generation, and movement of the robot arms. Then the component delivers the result in internal form to the answer-generation component.

- The **answer-generation component** transforms the internal results into a quasi-natural language form and interacts in a user-friendly manner with the casual user.

- The **knowledge base administrator** is responsible for the very important knowledge base and maintains, creates and changes the structure and the content of the knowledge base.

Fig. 21 shows the components of an expert system for a specific subfield of manufacturing.

Fig. 21: Functional units of an expert system



Japanese industry is aware that a software production technology is needed and that software is an industrial product.

In virtually all areas of technology, from the Space Shuttle to home electronics, the Japanese now see software as becoming even more vital than hardware and that improving productivity, reliability and maintainability have become extremely critical concerns. This is why, for example, NEC has established a laboratory for software product engineering to actively pursue R&D leading to production and management technologies for software.

Japanese industry is also concerned about the software life cycle. R&D activities in software production are probing virtually every step of the cycle, from basic research to evaluation of application results. Investigations are under way on requirement engineering, software design engineering, programming methodology, programming languages, automatic program generation, program verification, maintenance engineering, EDP (electronic data processing) auditing and even environmental engineering.

NEC's R&D activities and results reflect Japanese industry's efforts. NEC has recently come up with the software development and maintenance system (SDMS), a comprehensive system for supporting every facet of the development and maintenance of large- and medium-scale software. SDMS uses special techniques and aids to perform software design as well as software project and product management. NEC is also working on the development and evaluation of high-level programming language (HPL), which has superior descriptive power and more efficient code generation capabilities. This language is expected to reduce the time and manpower needed for large-scale software development and maintenance.

In another area of software production, NEC is working towards a complete, consistent, unambiguous specification that will describe what a software product will do. Efforts have included the study and development of specification languages, automated aids for documentation and analysis, and conversational graphic systems to help develop the specification.

The range of NEC's R&D activities in software management covers: evolving standardization technology, management aids, software quality control technology, and productivity measurement technology that will be widely usable. In addition, NEC is working towards vastly improved software project management through the use of EDP to perform the management now being done manually.

NEC efforts have yielded standardized technologies and engineering for programming support (STEPS), a standardization tool for development of business application software, and software quality control (SWQC), a means of maintaining high reliability, which is necessary if software is to become a viable industrial product. In fact, SWQC, as a small-group activity, is already at work throughout NEC and is proving to be highly effective for software development within the various sections of the company. As people are more motivated towards SWQC, the quality of software has improved markedly, and the "bugs" in some systems have already been reduced by half.

Human engineering is another chapter in the software development story, of course. At NEC, they are striving to provide a working environment where professional programmers can generate products efficiently. NEC is also seeking a software development environment where the power of the right types of computers - from ultra-large mainframe computers to microcomputers - is put to use in just the right places.

VI. ORGANISATION OF A SOFTWARE PRODUCTION UNIT IN DEVELOPING COUNTRIES

Scope and objectives of a software production unit

A software production unit, often called a software house, is a company dedicated to the development, installation, utilisation, maintenance and evaluation of software. Software comprises programs and programs packages running on hardware to solve various problems and tasks with the computer. In the past, mainly electrical engineers, mathematicians and physicians developed programs for users in a tailored way. Development costs for tailored software have become prohibitive due to the escalating salaries of highly skilled computer technicians, resulting in the so-called "software crisis". Efforts are under way to produce software in a more production-oriented manner, i.e., utilising software tools in the software production process and doing a tailored adaption only for special user requirements.

There are two major areas of focus in this endeavour. One side is technical- and scientific-oriented software production in R&D environments and in planning departments. Another is the immense number of programs that has to be developed in the commercial field. Some special subject areas are wages and salary accounting, order management, stock control, production scheduling and planning. Software houses have to support the user in all phases of the problem-solving process: fact-finding, requirements and restrictions, specifications, design, feasibility studies, economic analysis, design inspection, programming, testing, documentation, installation, evaluation, maintenance and tuning. The users have to be involved in the problem-solving process from the beginning. Software houses and consultant companies, as so-called third parties, play the mediator function between hardware companies and users. Normally they are closer and more important to the users than are the hardware companies.

Structure of a software production unit

As can be seen from the tasks described above, the staff of a software house have to undertake systems development work, on the one hand, and management functions on the other. In general, software houses have a range of only 5 to 50 employees; up to 300 to 500 is exceptional. As a rule, one-quarter of the staff is engaged in administrative and secretarial work, another quarter is partly occupied with management functions and half of the staff is devoted to consultancy work and software engineering problem-solving. A general manager or president and a vice-president form top management. Department heads responsible for several projects, together with project leaders responsible for individual projects, form middle management. The systems-oriented personnel hierarchy comprises the chief consultant, senior consultant, senior programmer, programmer and junior programmer.

A software company should be structured into four departments: software sales, software development, software quality assurance for mainframe and medium-sized computers, and customer support. Within departments, responsibilities should be distributed according to special commercial branches, subject areas within branches and types and different manufacturers of computers.

Management of a software production unit

Just like any other industrial firm, software houses should establish, at the end of each business year, a financial statement including a balance sheet. At the end of each business year, the financial plan, as well as cash flow and liquidity, and credit plan for the following year have to be prepared. Below is an example of distribution of costs.

Annual budget

Per Cent	Expenditure
54.5	personnel
5.0	rooms
5.0	promotional and legal costs
6.0	machines
4.5	office
5.5	taxes
7.0	miscellaneous
12.5	raw profits for investment, reserve, provision
100.0	total

This financial plan has to be updated quarterly according to the actual financial accounting statement of the previous quarter. Similarly, the projects plan has to be updated monthly or weekly according to the planning timetable.

Company standards for problem-solving, for project management and for all development phases of the software life cycle have to be set. Software tools should follow these standards so that employees adhere to them. These measures help guarantee that the produced software fulfills standards that ease the maintenance and updating processes of the programs after the installation phase, reduce software maintenance costs drastically, and prevent production of badly structured polluted software.

Staff training

To maintain their high-quality standards, the software products staff should participate in training courses regularly to update their knowledge about software engineering and production.

Important prerequisites in DCs

Prerequisites for effective functioning of software houses in developing countries are: good technical services by computer manufacturers for the installed hardware, operating system, file management or data base management systems. Furthermore, people from developing countries educated in computer science or related fields should, in addition, be trained in a software house or in a developed country.

Economic considerations

Industrialised societies are developing into information and service societies. Today's software houses will be the production units of the future. Because of lack of qualified personnel in developed countries and the escalating salaries of these experts, there is a great chance for fruitful co-operation between developed and developing countries. The raw profits of software houses as software production units could be 15 per cent of the revenue or even more. Investment costs are normally low as software development environments (SDE) are already available on personal computers and on personal systems. Such SDEs could be bought for anywhere between $30,000 to $50,000, hardware included. Developing countries have well-educated computer science and data processing people. If an official policy in this area already exists in a developing country, the way is paved towards co-operation. The example of Singapore in Chapter IV shows the feasibility of such an approach. The cost of starting a software house in a developing country is estimated to be less than $300,000. With support from the national government and UNIDO, this should be no problem. As salaries are much lower in developing countries, opportunities for entrepreneurs and venture capitalists in developing countries are much better than in industrialised countries.

VII. RECOMMENDATIONS ON BUILDING UP SOFTWARE PRODUCTION COMPETENCE IN DEVELOPING COUNTRIES

Since an overall government strategy for microelectronics and for application software development is normally missing in developing countries, a central authority close to the country's top office should be set up to handle this area. Because developing countries usually have inadequate telecommunications infrastructures, national strategies should not rely on telecommunications. Instead, strategies could be based on the following priority list:

- importing whole systems (hardware and software);
- importing hardware maintenance knowledge;
- importing application software development tools;
- production of components according to second-source licensing contracts with companies from industrialised countries supported by developing country governments;
- components design and manufacturing;
- whole systems design, realisation and evaluation;
- exporting software services; and,
- exporting hardware.

These steps can lead to increased confidence in the developing country products and indigeneous manufacturing. Awareness of software has to be heightened. Hardware-orientation has to be shifted to application software-orientation. Recycling of hardware and software should be planned.

Technology Transfer Centre

An education-oriented **Technology Transfer Centre** may be set up. It would be a forum of discussion for international experts and representatives of governments, research institutes and industry. Workshops, seminars, courses and meetings may could be initiated. The centre could support the combination of knowledge transfer and investment for application software and hardware systems technology. It could be supported by the government and international organisations. Control and support could be provided by the Ministry of Investment and the Ministry of Planning. Initially the center could be staffed by one research manager, two to four research and education assistants and one secretary. Some investment funds could go towards a textprocessing system and general microcomputer systems.

Centre for Microelectronic Applications and Software (MAS)

The government and international organisations could install, together with industry, a Centre for Microelectronic Applications and Software (MAS), which should have access to technical and business-related software information contained in international or national respositories. This centre and the proposed Technology Transfer Centre should be mutually supportive. The Technology Transfer Centre would concentrate on theory and education, the Center for Microelectronic Applications and Software on the practices and requirements of the industry. These two functions could also be merged under one institution.

Steps have been taken by UNIDO towards accelerating the development and growth of microelectronics and software industries in developing countries. Now, the need is for an international centre devoted to these industries. Such a centre would enable developing countries to compete in the world market, and would foster international co-operation for development and transfer of technology between developing and developed countries.

The creation of an international Centre for Microelectronic Applications and Software would help developing countries strengthen their technological infrastructure and capabilities in the field of microelectronics. Its aim would be to establish professional core groups in several developing countries focusing on concrete microelectronics applications (including product applications) and software production /Sc87/.

MAS will do this by providing developing countries with:

- technical services (advice on various problems, formulation of projects, documentation, and analyses;
- knowledge (on worldwide microprocessor applications, and software production and marketing);
- technology (in the form of software or readily applicable solutions); and
- information (on available microprocessor applications and software, software development methods, marketing, and training).

An example of assistance is in integrated software application. Firm A is able to design most of the components of a software system but lacks the ability to tie all the applications together in the system. MAS would arrange with firms B and C for software assistance and training in integrated systems design.

Participants in the activities of the centre would be industrial enterprises from developed countries, particularly computer and software manufacturers; governments and small- to medium-sized private companies, university and research institutions in developing countries; and worldwide professional associations and federations of industries.

MAS will train people to become system analysts and software engineers. These specialists, knowledgeable about computer systems and the problems to be solved, improve the reliability and quality of products. The centre would help create and strengthen indigeneous core groups of professionals in developing countries who are capable of undertaking software and product maintenance.

MAS offers potential advantages to developing countries that are at a disadvantage because of scant technical know-how in modern microprocessor applications and software production, little capital, limited choices in equipment, software, and services, and foreign exchange problems that prevent the acquisition of modern technology and capital equipment.

MAS would co-operate with private computer and software manufacturers to form multidisciplinary expert groups for the design, testing, maintenance, and marketing of microprocessor applications and software. These experts would also provide technical development and problem-solving assistance to developing countries.

UNIDO and other international organisations could support the establishment of a Technology Transfer Centre and the MAS as models for further investments in the microelectronics and application software development field in developing countries.

UNIDO and other international organisations could assist developing countries to set up a strategy for application software development through experts, seminars and conferences. Experts in modern application software development from industrialised countries could give seminars of 5 to 10 days in developing countries to raise the consciousness on polluted software and to widen knowledge on well-structured software. Suggested themes of seminars and conferences could be "importing software pollution into developing countries", "software production - a chance for developing countries?", and "how to develop structured software".

The MAS would bring together highly skilled people from universities, national research centres, research institutions, as well as marketing people and managers of small industries. For an initial three years, 50 per cent of the support to MAS should come from capital ventures and 50 per cent from the government and international organisations.

MAS should hire specialists in special application areas, hardware, software and information systems and operators and project managers. All phases of the problem-solving process should be supported by software tools imported from software houses in industrialised countries with whom effective co-operation should be maintained. The MAS management should be similar to the management of a normal industrial company with a balance sheet, statement of profit and loss, cash-flow review, project plans and annual financial plans.

Implementation of solutions (e.g., optimisation) on microcomputers in small industry as well as general support from the Technology Transfer Centre and the government will help the MAS become commercially independent after some years. When it has become more self-sufficient, the MAS could try to export software and services. The Technology Transfer Centre could also perform a publicity function to spread awareness of its work and to disseminate information to decision-makers. A group of experts, consultants and researchers should provide permanent support to the MAS.

Annex I

Selection Criteria for an Application Software Package

Experience in developed countries has shown that quality control and selection criteria for application software packages are crucial if serious problems are to be avoided. Besides the standard required characteristics of software products (easy to evolve, to maintain and to use; portable, reliable, robust and efficient) the answers to the following questions about application software packages are meant to serve as guidelines to their selection:

- How often is the application software package installed?

- How well described is its documentation (functional description, user handbook, system handbook, realisation handbook, installation handbook, test specification)?

- How well do its functions cover the required problem-solving? How big are the needed individual supplements? Is it possible to reach the objectives of the problem statement with the application software package?

- How well defined are its interfaces?

- How robust and reliable is it?

- Which parts are programmed for online handling and which for batch processing? Is the software package easy to use?

- Is it easy to change and modify?

- How high are the (unbundled) costs of the software package? Which costs are necessary for training, installation and individual adaptation?

- In how many system environments is the application software package installed? How portable is it?
- Is the run-time efficiency sufficient?
- Are milestones defined for the installation of complex application software packages?
- Which steps for design inspection and testing are planned? How detailed and specific are error messages?
- Does the application software package use a data base?
- Are there software tools for supplementing and changing the applications software package?
- Is it possible to specify authorisation hierarchies within the applications software package?
- Does the package have sufficient means for data protection and data security?
- How easily can cold and warm starts be executed?
- Which programming language is used?
- Which essential parameters of the package are restricted by which quantitative conditions?
- How often is it installed internationally?

Annex II

Criteria for Selecting a Softwarehouse

Experience in developed countries has shown that the selection of a stable, reliable and qualified software house is extremely important for an excellent data processing solution. The status of co-operation of a software house with an industrial company or governmental institution should be similar to the legal , tax and organisational consultants of that institution. The answers to the following serve as guidelines for selecting a software house:

- Is the software house sufficiently staffed (at least 20 employees)?

- Is it financially stable?

- Does it have enough positive current references?

- Does the software house fulfill cost and time schedule commitments according to customer statements?

- Are the qualifications of the employees up to standard?

- Does the software house possess the organisational and software competence and experience for the required problem area?

- Does it have a standard software package for the required problem area?

- Does it have enough experience with the required or existing systems environment (hardware, operating system, basic software, utility programs, tools, programming language)?

- Does it utilise sufficient software tools for all phases of the software development process?

- Does the software house utilise sufficient standards for individual software production phases of the software life cycle?

- Is it a member in a recognised computer science or consultancy society or association?

Glossary

Application Programmer (AP) - part of a software house or a data processing department of a company or institution; educated in computer science and data processing and able to implement data processing solutions.

Artificial Intelligence (AI) - area of computer science that develops concepts and methods for symbolic representation and processing of knowledge; systems are developed that simulate to some extent the cognitive behaviour of human intellectual processes.

Casual User (CU) - knowledgeable and well-educated user not familiar with data processing and computer science details; normally not willing to be instructed in the usage of a system longer than 10-20 minutes.

Data Base (DB) - set of structured data, together with a set of operations defined on these data, specifying the semantics of the data structure. Has an associated data base management system which guarantees consistent input, change, deletion, storing and output of data. Today contains so-called formatted data such as name, salary, manager, department.

Data Administrator (DA) - the person/function responsible for all information units of a company represented in form of data. In contrast to the data base administrator who is responsible for the efficiency and security aspects of the data base management system, the data administrator is responsible for the correctness and completeness of the data base.

Data Base Administrator (DBA) - the person/function responsible for the efficiency and the security aspects of the data base management system; has to have the overview of the global data type description (=schema) and has to build up and to maintain the schema.

Data Base Management System (DBMS) - a complex software system responsible for the integrity and management of data from the data base. Today's DBMS is embedded in a software development environment.

Multimedia Data Base (MMDB) - a structured set of formatted text, graphic, halftone and speech data; very important in future office automation and computer integrated manufacturing application systems. Also called non-standard data bases.

Multimedia Data Base Management System (MMDBMS) - a complex software system responsible for the integrity and management of data from a multimedia data base. Controls the consistent transition of one state of a multimedia data base into another.

Software Development Environment (SDE) - a set of software tools for the specification and generation of software products. Guides an application programmer in designing layouts of screens and lists, helps to specify consistent dialogs and to gain structured programs.

Software Product (SP) - a piece of program code written in a higher programing language like ALGOL, BASIC, C, COBOL, FORTRAN, PASCAL or PL/1 or in an assembly language. Today's software products are often realised with software development environments. A software product is often installed in many companies or other institutions. Normally it is marketed and sold by software houses.

Software House (SWH) - a software production unit that develops, maintains, markets and sells software products and undertakes individual adaption of these products to the specific requirement of companies or other institutions.

..:ferences

A. UNIDO Publications

/Co85/ Correa, C.M.: Trends in commercialization of software in developing countries, UNIDO/IS.574, 7 November 1985

/Esun85/ ESCWA and UNIDO secretariats: Silicon foundry and design centres in the Arab region: issues and approaches, UNIDO/IS.583, 12 December 1985

/Ko84/ Kopetz, H.: Guidelines for software production in developing countries, UNIDO/IS.440, 10 February 1984

/Mi84/ Miles, T.O.: A competitive assessment of the U.S. software industry, U.S. Department of Commerce, December 1984

/Na84/ Narasimhan, R.: Guidelines for software development in developing countries, UNIDO/IS.439, 10 February 1984

/Na85/ Narasimhan, R.: The UNIDO programme on microelectronics: an analytical perspective, UNIDO/IS.529, 24 April 1985

/Sc87/ Schware, R.: Proposal for an International Center for Microelectronic Applications and Software (MAS), UNIDO Internal Working Paper, 1987

/So86/ Soltysinski, S.: UNIDO/REMLAC preparatory report on the elaboration of a programme of action at strengthening the negotiating capabilities in the acquisition of hardware and software, UNIDO Internal Working Paper, 4 March 1986

/So87/ Soltysinski, S.: Strengthening negotiating capabilities in the acquisition of hardware and software in Latin America, UNIDO Report IPCT.15, 18 February 1987

/Un83/ UNIDO secretariat: Problems of software development in developing countries, UNIDO/IS.383, 27 April 1983

/Un85/ UNIDO: Survey of government policies in informatics, UNIDO/IS.526, 4 April 1985

/Un8€/ UNIDO: The UNIDO programme on technological advances, UNIDO/IS.411/Rev.2, 27 October 1986

B. Other Publications

/Al78/ Alford, M.W.: Software Requirements Engineering Methodology (SREM) at the Age of TWO, Proc. COMSAC 78, 332-339, 1978

/Be81/ Bever, M.; Gombe, W.; Lockemann, P.C.; Mayr, H.C.: Method Base Systems: A Tool for Decision Support, Univ. Karlsruhe, Proc. Nato Advanced Study Institute 1981, Lisboa

/Bo82/ Bonner, S.; Shin, D.G.: Comparative Study of Robot Languages. COMPUTER 5(12), 1982

/Bu80/ Bubenko, J.A.jr.: Information Modeling in the Context of System Development, Proc. IFIP Congress 1980, North-Holland, 1980

/Ca81/ Carlson, E.D. et al.: Display Generation and Management Systems (DGMS) for Interactive Business Applications, Vieweg 1981

/Ch80a/ Cherniavsky, V.S.; Konrad, E.: On Limitations for Algorithmic Modeling of Intelligent Activities, Proceedings of the Conference on "Artificial Intelligence and Information-Control Systems of Robots", Bratislava (CSSR), 1980

/Da77/ Davis, C.G.; Vick, C.R.: The Software Development System, IEEE-Transactions on Software Engineering, 69-84, 1977

/Di84/ Diebold Management Report No. 8/1984

/En80/ Engel, H.H.; Schneider, H.-J.; Stübel, G.: AWUS - an Aid to Working with and Understanding of (Application) Software, in: Eberth, et al. (eds.): Practice in Software Adoption and Maintenance, North-Holland Publ. Co. 1980

/Go80/ Godbersen, H.P.: On the Problem of Time in Nets, First European Workshop on Petri Nets and related System Models, Straßburg, 23-29.9.80, in: Girault, C.; Reisig, W. (eds.): Application and Theory of Petri Nets, 23-30, 1982

/Go80a/ Godbersen, H.P.; Meyer, B.E.: A Net Simulation Language, Proceedings Summer Computer Simulation Conference 1980, Seattle, August 25-27

/Ha80/ Habel, Ch.; Rollinger, C.-R.; Schmidt, A.; Schneider, H.-J.: A Logic-oriented Approach to Automatic Text-Understanding in: Bolc, L. (ed.): Natural Language Based Computer Systems, Hanser-Verlag, 57-117, 1980

/Ham79/ Hamilton, M.; Zeldin, S.: The Relationship between Design and Verification, The Journal of Systems and Software 1, 29-56, 1979

/He80/ Hewlett-Packard: Using V/3000 - an introduction to forms design for non-programmers, Part/No. 32209-90004, Cupertino, 1980

/IBM74/ IBM Corp.: HIPO (Hierarchical Input-Process-Output) - a Design Aid and Documentation Technique, IBM GC 20-1851, 1974

/IN86/ INPUT, LTD: Major Western European markets for information services - analysis and forecasts, 1986-1991, London, November 1986

/Ja77/ Jackson, M.: Data Structure as a Basis for Program Design, in: Bated, D. (ed.): Software Engineering Techniques, State of the Art Report, Infotech, 1977

/Kon81/ Konsynski, B.R.: Model Management in Decision Support Systems, Proc. Nato Advanced Study Institute 1981, Lisboa

/Le79/ Leavenworth, B.: A Data Flow Pseudocode, Technical Report RC 7772, IBM Thomas J. Watson Research Center, 1979

/Li79/ Linstone, H.A. et al.: The Use of Structural Modeling for Technology Assessment, Technological Forecasting and Social Change 14, 291-327, 1979

/Lo81/ Loomis, M.E.S.: The Nature of Database Management for Effective Decision Support Systems, Proc. Nato Acvanced Study Institute 1981, Lisboa

/Lu79/ Lundeberg, M.; Goldkuhl, G.; Nilsson, A.: A Systematic Approach to Information Systems Development, Information Systems 4, 1-12, 93-118, 1979

/Met78/ Methlie, L.B.: Information Systems Design, Universitetsforlaget, Bergen, Oslo, Tromso, 1978

/Me75/ Meyer, B.E.; Schneider, H.-J.: Predicate Logic and Data Base Technology, Advanced Course on Data Base Languages and Natural Language Processing, Freudenstadt, 1975

/Me79a/ Meyer, B.E.; Schneider, H.-J.: Tools for Information Systems Design and Realization, in: Schneider (ed.): Formal Models and Practical Tools for Information Systems Design, Proc. of the IFIP WG 8.1, Oxford 1979, North-Holland, p. 1-29, 1979

/Me81/ Meyer, B.E.: COLAN - A Language for the Communication between Simulation Models, SCSC 1981

/Mo82/ Moto-Oka, T. (ed.): Fifth Generation Computer Systems, North-Holland Publishing Company, 287 pages, 1982

/Mu79/ Munz, R.; Schneider, H.-J.; Steyer, F.: Application of Subpredicate Test in Database Systems, in: Proceedings of the VLDB, Rio de Janeiro, IEEE, 1979

/No81/ Nof, S.Y.: Theory and practise in decision support for manufacturing control, Proc. Nato Advanced Study Institute 1981, Lisboa

/Nu75/ Nunamaker, J.F.; Konsynski, B.R.: From Problem Statement to Automatic Code Generation, Systemeering 75, 215-240

/Pe73/ Petri, C.A.: Concepts of Net Theory, Proc. Symp. on Math. Found. of Comp. Science, High Tatras, 1973

/Ra74/ Rabin, M.O.: Theoretical Impediment of Artificial Intelligence, in: IFIP Congress 1974, North-Holland, 1974

/Ro77/ Ross, D.T.: Structured Analysis, A Language for Communicating Ideas, IEEE Trans. on SE, Vol SE-3, No. 1, pp. 16-34

/Ro83/ Rollinger, C.-R.; Schneider, H.-J.: Text Understanding as a Knowledge-based Process, in: IRFIS-Proceedings, Heidelberg, 1983

/Sa82/ Salton, G.; Schneider, H.-J. (eds.): Research and development in information retrieval, Proc. of the Fifth International Conference on Information Retrieval, Berlin, May 1982, Springer Lecture Notes in Computer Science, 321 pages, 1982

/Scn80/ Scheschonk, G.: On the Construction of System Nets. European Workshop on the Application and Theory of Petri Nets, Straßburg 23.-29.09.80, in: Girault, C.; Reisig, W. (eds.): Application and Theory of Petri Nets, 104-108, 1982

/Scn76/ Schneider, H.-J.: On Planning and Implementing "Intelligent Systems", Semantic Network Project No. 1/76, Techn. Iniv. Berlin, 10 pages, 1976

/Scn79/ Schneider, H.-J. (ed.): Formal Models and Practical Tools for Information Systems Design, Proc. of the IFIP-WG 8.1, Oxford 1979, North-Holland, 306 pages, 1979

/Scn80/ Schneider, H.-J.: Sind computergestützte Informationssysteme bei zunehmender Komplexität noch beherrschbar? Proceed. 6. Internationaler Kongreß Datenverarbeitung im europäischen Raum, Wien, 17.-21.03.80, ADV Band 1, 741-755, 1980

/Scn82a/ Schneider, H.-J.; Wasserman, A.I. (eds.): Automated Tools for Information Systems Design, Proc. of the IFIP-WG 8.1 Working Conference, New Orleans, January 1982, North-Holland, 268 pages, 1982

/Scn82b/ Schneider, H.-J.: Techniques and Formal Tools for Design, Realization and Evaluation of Evolutionary Information Systems. In: Hawgood, J.: Evolutionary Information Systems, Proc. of the IFIP-TC8 Working Conference, Budapest, September 1981, North-Holland, 13-37, 1982

/Scn82c/ Schneider, H.-J. (ed.): Distributed Databases, Proc. of the Second International Symposium on Distributed Databases, Berlin, September 1982, North Holland, 375 pages, 1982

/Scn84a/ Schneider, H.-J.: An Integrated Concept for Information System Development, Proc. CAE Symposium at Jiatong University Shanghai, 1984

/Scn84b/ Schneider, H.-J.: Software der fünften Generation - die Lösung der Softwarekrise? - In: Wissenschaftsmagazin der TU Berlin zum Thema "Kommunikation", Heft 6, 80-84, 1984

/Scn86a/ Schneider, H.-J.: Systems of the 5th Generation and their Impact on the Qualification, Training, and Retraining of Management, Proc. Int. Conference on Artificial Intelligence in Manufacturing, Duttweiler Institut, Rüschlikon, Switzerland, November 1985

/Scn86b/ Schneider, H.-J.: Communication in Automotive CIM Environments using Cumulates, 15th Int. Symposium on Automotive Technology and Automation, Flims, Switzerland, October 1986

/Scn86c/ Schneider, H.-J.: The VDN/RDS/REFLEX Relational Database System, in: Pergamon Infotech State of the Art Report on "Relational Databases", 195-209, 1986

/Scn86d/ Schneider, H.-J.: Formale Gestaltungsaspekte in der Systementwicklung, in: Handbuch der modernen Datenverarbeitung, Heft 130, 3-17, Forkel-Verlag, 1986

/Scn86e/ Schneider, H.-J.; Rastgooy, K.; Trümner, H.: Qualitätssicherung von CAD-Scftware durch netzorientierte Spezifikation, CAD Kolloquium, November 1986, Sonderforschungsbereich 203, TU Berlin

/Se80/ Sernadas, A.: Logical Procedure Definition for Information Systems Specification, Thesis 1980, L.S.E. Univ. of London

/Stu80/ Studer, R.: A Dialogue Interface for Data Base Applications, Proc. Very Large Data Bases, 167-182, Montreal, Canada, 1980

/Was80/ Wassermann, A.I.: Information System Design Methodology, Journal of the American Society for Information Science, 5-24, 1980

/Was81/ Wassermann, A.I.; Shewmake, D.: Automating the Development and Evolution of User Dialogue in an Interactive Information System. In: Hawgood, J.: Evolutionary information systems, Proc. of the IFIP TC8 Working Conference on Evolutionary Information Systems, Budapest 1981, North-Holland, 1982

/Win86/ Winter, D.; Scheschonk, G.; Gier, K.: ISAC - ein System zur Unterstützung der Systembeschreibung und der Systemanalyse, in: Handbuch der modernen Datenverarbeitung, Band 130, 97-106, Forkel-Verlag, 1986