The Department of Computer
and Information Science
Linköping University

Annual Research Report 1984

LiTH-IDA-R-84-21 December 1984

The Department of Computer and Information Science

Linköping University

Annual Research Report 1984

This report describes research on software (and to some extent hardware) technology within the Department of Computer and Information Science at Linköping Institute of Technology, which is a part of Linköping University. Main areas of current research are programming environments, artificial intelligence, application systems, computer-aided design of digital systems, library and information science, and theoretical computer science. The department has a well organized integrated program for graduate studies (PhD and Licentiate degree), with a large faculty engaged in research and thesis supervision. In addition to the research organization and the extensive undergraduate course program, there is also a knowledge transfer program at the department where we cooperate with large Swedish companies on medium to long term R&D issues.

Mailing address:
Dept. of Computer and Information Science
Linköping University
S-581 83 Linköping
Sweden
Tel: int + 46 13 28 10 00

Postadress:
Inst. för datavetenskap
Universitetet och
Tekniska Högskolan i Linköping
581 83 Linköping
Tel: 013 - 28 10 00

CONTENTS

1.

Introduction

1.1 Organization of the department.

The Department of Computer and Information Science was formed in 1983. It has presently about 80 employees (about 15 with a PhD) and activities are divided approximately equally between research and teaching. The research and graduate education program is organized in common for the whole department, but research projects are carried out within smaller groups, research laboratories, which typically consist of five to ten persons. The undergraduate teaching encompasses the three subject areas computer science (*datalogi*), telecommunications and computer systems (*telesystem*) and administrative data processing (*administrativ informationsbehandling*).

The research program is coordinated by the board for research activities and graduate education, headed by Erik Sandewall. The current laboratories are PELAB (Lennartsson) for programming environments, ASLAB (Hägglund) for application systems, AILAB (Hein/Tengvald) for artificial intelligence, CADLAB (Lawson/Lyles) for computer-aided design of digital systems, and LIBLAB (Hjerppe) for library and information sciences. In addition there are special research groups for theoretical computer science (Maluszynski and Lingas), which serve as a shared resource for the laboratories. The group for administrative data processing, although primarily a group for undergraduate teaching, also includes some research activities.

The major funding for our research is supplied by the Swedish Board for Technical Development, STU, (84/85: 5 MSEK) In addition to the STU support, funds are also provided by the Delegation for Technical and Scientific Information Supply (DFI, 84/85: 0.75 MSEK), NFR (84/85: 0.075 MSEK), other sources (84/85: 0.35 MSEK) and the ordinary university budget for research and postgraduate studies (84/85: 2.3 MSEK). The joint program for knowledge transfer to industry has an additional budget (84/85: 2.1 MSEK), provided by the participating companies. At present 1 SEK is approximately 0.11 USD.

This report is a slightly revised edition of the 1983/84 progress report issued to satisfy the requirements for the largest research grant, from STU. The style of presentation is thus somewhat biased towards the needs of readers who have followed the development during the five year period 1980-1985, when the national program for knowledge development in information processing has been in effect. In this edition of the report, we have however tried to make the presentations suitable for a wider audience and we have also added descriptions of research not funded by that STU grant.

After the main body of this report was written, a few organizational changes in the research program have been made. A new group has been formed, mainly from the office systems group in ASLAB, now called *The Laboratory for Representation of Knowledge with Logic.* This group is headed by Erik Sandewall and the main topics to be studied are non-monotonic logic, truth/reason maintenance, probabilistic reasoning, reasoning about time and plans, theory of information management systems, databases and logic. There are also plans to organize a joint research group with some other departments in the area of Robotics.

1.2 Objectives of the research programmes.

The major part of the research is currently funded from the Swedish Board for Technical Development, STU, under the five-year programme for knowledge development in information processing and the national programme for microelectronics. These programmes emphasize competence build-up, development of excellency in selected areas but also that results from the work should be transferred to applications during the contract period. Applications could be found in computer industry, commercial users of computer systems, public administration, or in other areas of research.

These goals are sometimes competing or contradictory. We have tried to balance our efforts so that the different goals would be achieved equally well. We will detail, in later sections, how work during the period so far has resulted in 10 Ph.D. theses and 46 internationally published research papers. We will also detail how active cooperation and knowledge transfer is being pursued with several large Swedish corporations (in particular, Ericsson, ASEA, and S-E-banken), and how several spin-off companies have been formed from our group.

As it turns out, the actual state of our department at this time agrees reasonably well with the development goals stated by STU: the number of available teachers and advisors for graduate study (or in other words, researchers and teachers with a PhD degree) has grown to 14 against 4 in 1980, and they together cover a good part of the specialities within computer science. Also important, the organization of the department is now much better developed than it was in 1980.

In the remainder of this introductory chapter, these general development goals and the steps to achieve them will be discussed in some more detail.

1.3 Build-up of Research Areas.

The research proposal for the STU supported knowledge development activities was written in 1979, five years ago, and proposed research in two areas: *programming environments* and *application systems*, the latter including the study of application modelling, dialogue systems, and office applications. Additional proposals were made later for a *laboratory, for artificial intelligence*, with an emphasis on expert systems and for an *LSI Design Center* in

cooperation with the Physics department, one part of which has evolved into the current CADLAB.

Looking in retrospect, we observe that these were fortunate selections, and that all these named areas are considerably more 'popular' now than five years ago. From the reports for the respective laboratories, we can also see that their initial plans have provided useful guidance.

But as could be expected, we have also seen additional research areas during this period that also were well worth covering, and which connect organically to the existing areas of study. To some extent those new fields have been assimilated within the existing laboratory structure:

- *formal specification methods* in the programming environments laboratory;

- *functional programming* also in the programming environments laboratory (Pär Emanuelsson);

- *statistical information systems* in the application systems laboratory (Bo Sundgren).

In other cases the connections have been established by organizational means:

- in 1983, the new Department of computer and information science (IDA) was formed from the previous datalogi, telesystem, and ADP (Administrative Data Processing) groups. This has led to rapidly increasing interactions between the C.A.D. laboratory, where computer architecture and VLSI design techniques are studied, and the other groups;

- also in 1983, we joined the inter-Nordic SYDPOL (System Development environment and Profession-Oriented Languages) project, in cooperation with researchers from the universities of Oslo, Aarhus, and Stockholm in the areas of user interactions, and the effects of information systems on user milieu;

- during 1983 and 1984, courses and research in the area of office systems has been strengthened through cooperation with office systems research led by Jan-Olof Brüer in the Electrical Engineering department.

Finally, new areas of activities and new groups have been formed, under the leadership of arriving researchers:

- *logic programming* and *attribute grammars*, started by Jan Komorowski and continued with additional breadth by Jan Maluszynski;

- *geometrical complexity*, started by Andrzej Lingas, later joined by Christos Levcopoulos;

- *library information science*, in the laboratory started more recently by Roland Hjerppe.

Through all of these means, we now have a research environment with considerable diversity, where at the same time the existence of a joint department and a laboratory structure provides the cohesion or 'glue'. In this

milieu, the graduate students are exposed to a multitude of research specialities, so that they can make an informed decision about which area to choose for themselves, and where interactions between specialities is an everyday reality. Appendix 2 presents a list of the available advisors for the graduate students in our department, and their background and present research interests.

One significant aspect of the departmental build-up is that we are now able to offer a set of courses for graduate study. Both the necessary requirements for such courses (teachers, and students) are present now to a much larger extent than five years ago. In particular, the closer cooperation with the other groups and industries listed above, have contributed to that student base. Appendix 2 shows the courses that are offered during the present academic year.

Another significant aspect of the department is that the base of computer equipment has been strengthened, through the addition of six Xerox Lisp-machines on an Ethernet, as well as by the gradual extension of our previous DEC-oriented system (DEC-20, PDP-11:s, DECNET). The existence and reliable operation of this base has been significant, not only for our own work, but also for the knowledge transfer. See also appendix 5.

These things, taken together, represent the results of our efforts to build up a viable research milieu and at the same time provide the basis for high-quality undergradate and graduate study programmes in computer science.

1.4 Knowledge Transfer Activities

1.4.1 Improvements in Undergraduate and Masters-level Teaching

In the long range, the most significant method for knowledge transfer is through undergraduate and masters-level education. The development of our research programme has contributed to that education in several ways.

First, a new computer science 'line' (*datavetenskapliga linjen*) has been started in addition to the computer science and engineering curriculum (*datateknik-linjen*). This new line is in the school of engineering, but differs from ordinary engineering curriculums (such as electrical engineering, or mechanical engineering) in some significant ways:

= significantly more discrete mathematics, partly gained by reduction of the calculus courses

= courses in theoretical branches of computer science

= courses in AI and AI-oriented subjects

= Lisp as the first programming language

= relevant humanities, such as psychology and linguistics, are significant parts of the curriculum.

We accept 30 students per year to this curriculum, and the first set are now in their third year (out of four years). It is already quite clear that these students develop a different 'culture', and in particular a more solid basis for graduate research in computer science, than what students in our other lines do. While certainly our other lines will continue to be of very high importance, the computer science line has provided a significant addition.

Second, the set of courses that are available in the other lines has been extended, and many of the courses have been improved. Technically, this has often been done by making new courses from the computer science line available to other lines as well, but it is the STU funded research that has provided the competence base for the new courses. In the computer science and engineering line, a specialization for telematics has been added, relying partly on our research in interactive systems and office systems.

Thirdly, the mechanical engineering line has been extended with a new specialization that combines mechanical and computer engineering. We believe that especially the STU-funded research in artificial intelligence will be significant within that specialization.

Details about these curriculums and the set of courses there are given in appendix 3.

1.4.2 The Knowledge Transfer Program

Knowledge transfer via undergraduate education is efficient in the long run, but slow to take effect. We have instituted a *knowledge transfer program*, KTP, together with a limited number of industries:

ASEA
Ericsson
S-E-Banken

(a few more may be added). The goal of KTP is to 'inject' competence derived from research into the existing industrial organization. The method is that at least one person, located on a middle level in the organization, comes to our department for a period of one or a few years, in order to learn new technology, and returns to his organization after that time. The participating company also pays a yearly contribution that helps pay for researchers (particularly guest lecturers) and equipment.

More details about KTP are given in appendix 1.

1.4.3 Spinoff Companies

The significance of university spinoff companies for industrial growth is well known. A considerable part of our artificial intelligence laboratory, lead by Uwe Hein, split off in the spring of 1984 and formed Epitec AB. The new company has presently 10 employees, of whom two are part time consultants from our department. The main effort goes into development of a commercial

product for building expert system based on experiences from the AILAB. The company is also engaged in consulting and is presently assisting several Swedish companies in the development of knowledge-based systems.

A few years ago, Jerker Wilander and Kenth Ericson founded the company Softlab AB in Linköping. Softlab is working in the area of compiler design, and they have developed the front end for the PLEX compiler now used at LM Ericsson.

Recently Grafitec AB has been founded by Michael Pääbo and others from the CADLAB group. Grafitec will be active in business graphics and, later on, in scene animation.

Some other spinoff companies in Linköping have required a considerable number of software personnel, although their main business is something else. In particular, Context Vision (formed in 1983, for building picture processing systems) has recruited heavily from our department.

2.

PELAB
The Programming Environments Laboratory

The research in PELAB is a continuation of the work in program manipulation projects, that was done here during the seventies, and of the INTERLISP experience in general. When PELAB was established in 1980 the concept *Programming Environment* was a novel one to most people. Since then the area has been recognized both for its industrial relevance and as a research topic.

2.1 Researchers and Projects

2.1.1 Major events

The DICE project has been the 'kernel' project and thus the main activity in PELAB during this year. Peter Fritzson's doctorate dissertation, Johnny Eckerland's licentiate thesis, as well as several master theses, have been presented during 1984 and represent the progress in the DICE project.

A member of PELAB, Christos Levcopoulos, has gradually shifted to join Andrzej Lingas' geometrical complexity research, and has obtained very good results there.

2.1.2 Laboratory Members

	% 84/85	% 83/84
Laboratory leadership:		
Bengt Lennartsson, PhD	75	75
Gunilla Lingenhult, secr	30	35
Supervisors:		
Pär Emanuelson, PhD	25	25
Anders Haraldsson, PhD	10	10
Peter Fritzson, PhD	80	

Employed graduate students:

Rober Bilos	75	50
Johnny Eckerland	85	50
Kristina Ernstsson	75	75
Peter Fritzson		80
Mariam Kamkar	75	50
Christos Levcopoulos		75
Nahid Shahmehri	75	50
Dan Strömberg		20
Lars Strömberg	50	

Research engineers:

Ralf Nilsson	40
Ola Strömfors	20

Associated personnel:

Kenth Ericson, Softlab AB, Linköping
Kjell-Håkan Närfelt, University of Luleå
Jan Komorowski, Harvard University
Dick Schefström, University of Luleå
Jerker Wilander, Softlab AB, Linköping

2.2
DICE, Distributed Incremental Compiling Environments.

(Bilos, Eckerland, Ernstsson, Fritzson, Kamkar, Shahmehri)

The present kernel project in PELAB is DICE (*Distributed Incremental Compiling Environment*). We are aiming at the development of an appropriate architecture for a full scale integrated environment supporting the development of programs coded in block-structured languages.

In the prototype of DICE the tools are running on a DEC-20, the host, where also all the information of the developed program is saved. The host is connected to a target, a PDP-11, where the developed program is executed. Among the results should be mentioned that *the flexibility normally available in an interpreting system could be achieved in a compiling system also*, and that *the functionality of a high level target debugger can be obtained via the incremental compiler without any target code instrumentation*, and without the existence of a target debugger at all.

We have gained experience about how to generate and integrate the traditional language-oriented tools like structure editor, parser, pretty-printer, code-generator, debugger, *etc.* The experience, in short, is that *the total efforts to make these tools integrated and incremental were much less than could be expected.* We have drawn the conclusion that incremental algorithms, in this case, do *pay for themselves purely by their efficiency improvements.* In addition, the support for experimental software development is available and could be used at will, free of charge.

2.2.1 DICE - Basic ideas

It is becoming more widely recognized that improvements in programmer productivity may be achieved if the programming environment is interactive and incremental, and if the various tools in the environment are integrated, i.e. they interact with each other in a way that amplifies each others capabilities.

The DICE system (Distributed Incremental Compiling Environment) strives to achieve these goals under several constraints. The system should be able to operate on compiled code and the developed program should be kept separate from the development environment. Some of the more important points of the DICE system are cited below.

- *Remote debugging and maintenance* is easy to achieve with the DICE system configuration.

- An incrementally compiling system like DICE which has a program data base is especially suitable for the *development of big programs*, on the order of 20000 to 100000 lines of code. Compiled code gives fast execution and incrementality gives fast program update and powerful debugging facilities.

- *Separability* - the compiled program is separated from the source code so that it can execute outside of the program development system.

- *Connectivity* - the DICE system can be connected to a malfunctioning production program in order to debug it or to correct it. This can be done after the error has occurred and need not be planned in advance.

2.2.2 Comparison with related work

A number of incremental programming environments have been developed during the last decade. Examples are: INTERLISP, the Cornell Program Synthesizer, GANDALF, COPE, PATHCAL, ECL, and LISPEDIT.

Though some of these systems, e.g. INTERLISP, GANDALF, ECL and LISPEDIT, support incremental compilation, they usually take procedures as minimal compilation units. Statement-level incremental compilation has more often been employed for line-oriented languages like BASIC, and very seldom for languages with nested statement structure. However, the architecture of the DICE programming environment makes it possible to change single statements or single declarations in a PASCAL program without unnecessary recompilation of whole procedures, as is described in more detail later in this paper. This capability requires that the system keeps a record of machine code positions of statement boundaries. The availability of such information also simplifies the implementation of debugging facilities such as single-stepping. Mitchell [Mitchell-79] gives a thorough discussion of certain aspects of the design of interactive programming systems, especially the consequences of allowing a close mixture of interpreted and compiled code at a granularity which may be finer than elementary statements.

We summarize new or unusual features in the DICE system:

* Traditionally, program development systems which support debugging and testing in embedded systems have not been incremental. Such systems often operate in a host-target configuration. The DICE system is to our knowledge the first fully incremental programming environment for a high-level language, which supports this kind of system development.

* Statement level incremental compilation is carried out on a full-sized language with nested statement structure (PASCAL).

* Incremental recompilation is possible even after changes to global declarations.

* The incremental properties and the debugging capabilities of the DICE system are based solely on compilation. This principle of implementation is in agreement with the GANDALF implementation but is in contrast to most other incremental systems, which partly or exclusively rely on interpretation.

* The DICE system maintains a cross-reference and static-analysis database as an integral and necessary component. Some DICE features are critically dependent on this database, e.g. incremental recompilation after changes to global declarations, and some facilities in the debugger. The MASTERSCOPE subsystem in INTERLISP is perhaps the most well-known example of a similar tool. MASTERSCOPE is implemented as a resident relational database, and may be interrogated through a query language. The DICE cross-reference database is even more integrated into the system. DICE cannot function without it, and the cross-reference information is stored as a part of the symbol table. The symbol table has been partitioned so that DICE can accommodate large programs. There are two reasons to integrate cross reference information into the symbol table. First, the incremental compiler needs fast access to both the symbol table and the cross reference information. Second, the cross reference information will automatically be partitioned by the symbol table partitioning mechanism.

* The language-oriented editors generated by the DICE system are hybrid text- and structure editors, which gives the user maximal flexibility. Parsing is automatically performed according to the subgrammar which correspond to the syntactic category of the current subtree. In contrast, the language-oriented editors generated by GANDALF system [Medina-Mora-82] are pure tree editors. However, their work has been a great inspiration to us. The current design has also been influenced by the hybrid editors of the PATHCAL system [Wilander-80] and the Cornell Program Synthesizer together with extensive experience from the INTERLISP tree editor, the EMACS text editor and the ED3 text editor for tree-structured documents [Strömfors,Jonesjö-81].

2.2.3 DICE - System Overview

The DICE command level has been designed to be as mode-free as possible. Debug commands, expressions and statements to be evaluated in the target computer, and queries to the cross reference data base may all be entered at the top level command loop. The editor may be used on the commands themselves or on a program unit such as a procedure.

Programs in the DICE environment are internally represented as abstract syntax trees. Various tools such as the screen-oriented structure editor, the incremental compiler, the prettyprinter and the static analyzer operate on the abstract syntax.

Debugging commands are entered at the normal DICE command level, and all legitimate PASCAL statements are part of the command language. The debugger is machine-independent - it calls the incremental compiler which generates code for evaluation of commands, or modifies the machine code of the target program for insertion of breakpoints etc. Essentially all machine-dependencies are isolated inside the code generator of the incremental compiler.

The system performs statement-wise incremental compilation, which makes it particularly useful to the debugger which operates at the statement level. Statement level recompilation usually is an order of magnitude faster than procedure-wise recompilation, which is important for big programs or on loaded time-sharing computers. Also, this permits acceptable interactivity even after changes to certain global declarations. The extra information needed in the program database to support incremental compilation is almost identical to that needed for debugging, which is an example of the economy that is possible in an integrated system. The system allows continued execution after most program changes.

Program editing is performed with a full-screen structure editor. The editor marks new or changed nodes in the tree of the current procedure. Recompilation of changed statement nodes is performed during a preorder traversal of the tree. New and old machine code is merged, and branch instructions in the old code are updated. Branch updating can be incremental on the statement level for languages with well-formed control structures on machines with PC-relative goto instructions.

Incremental compilation is consistent, i.e. no degeneration of code quality occurs after many edit-recompilation cycles, see [Fritzson-84b]. No global optimizations over statement-boundaries are performed by the compiler. Dynamic linking is facilitated by generating position-independent code and having an extra level of indirection for procedure calls.

Most information in the data base is packaged in procedure units. Each unit contains the source tree program representation, machine code, local symbol table and information from static analysis such as cross reference information.

In a Distributed Incremental Compiling Environment the host computer must have complete control over the target computer. It must be able to stop and start execution on the target machine and it must be able to manipulate code

and data inside the target.

2.2.4 Transformation of the prototype into a production version

A transformation of the DICE prototype from INTERLISP to a PASCAL implementation was planned from the very beginning. A PASCAL like subset of INTERLISP has been used, and in a master thesis work a transformation tool has been developed. The transformation is now underway, and the new host will be a SUN-2 work station. This will eliminate the current address space problems on the DEC-20, and we will also have the powerful user interface of the SUN available. We hope to have DICE running on SUN during the second quarter of 1985.

The transformation is motivated both from the research and from the relation-to-industry point of view. When the transformation has been completed DICE will be a portable environment. The host-target concept can be applied to the migration of DICE itself. The machine dependence of the whole system is limited to parts of the code generator of the incremental compiler, and to a few very small target primitives. Less than two hundred bytes of machine code have to be in the target originally in order to make a connection to DICE possible.

Part of the transformation will be done in PELAB, and in parallel Peter Fritzson will spend a year and a half with SUN Micro Systems and work on the next generation of their environment.

2.2.5 Additional tools

Licentiate theses projects on interactive tools for static analyses (*Mariam Kamkar, Nahid Shahmehri*), and on version control (*Kristina Ernstsson*) in DICE have been initiated. A language independent mechanism for error diagnosis and recovery (*Rober Bilos*) will also be included.

2.2.6 References

[Eckerland-84] J. Eckerland, *Retargeting of an Incremental Code Generator*, Licentiate Thesis, Department of Computer and Information Science, Linköping University, Linköping, Sweden; Nov. 1984.

[Fritzson-82] P. Fritzson, *Fine-grained Incremental Compilation for PASCAL-like Languages*, LiTH-MAT-R-82-15, Software Systems Research Center, Linköping University; July 1982.

[Fritzson-83a] P. Fritzson, *A Systematic Approach to Advanced Debugging through Incremental Compilation*, Proc ACM SIGSOFT/SIGPLAN Software Engineering Symposium on High-Level Debugging, in SIGPLAN Notices. Vol. 18, No 8, Aug 1983.

[Fritzson-83b] P. Fritzson, *Adaptive Prettyprinting of Abstract Syntax applied to ADA and PASCAL*, LITH-IDA-R-83-08, Department of Computer and Information Science, Linköping University, Linköping, Sweden; Sept 1983.

[Fritzson-84a] P. Fritzson, *Symbolic Debugging through Incremental Compilation in an Integrated Environment*, to appear in the Journal of Systems and Software, spring 1984.

[Fritzson-84b] P. Fritzson, *Towards a Distributed Programming Environment based on Incremental Compilation*, PhD Thesis, Department of Computer and Information Science, Linköping University, Linköping, Sweden; April 1984.

[Medina-Mora-82] R. Medina-Mora, *Syntax-Directed Editing: Towards Integrated Programming Environments*, PhD Thesis, Dept. of Computer Science, Carnegie-Mellon University, Pittsburgh, Pa. 15213.

[Mitchell-70] J. G. Mitchell, *The Design and Construction of Flexible and Efficient Interactive Programming Systems*. Ph.D. Thesis, Carnegie Mellon University. Reprinted by Garland Publishing, Inc. New York and London, 1979.

[Strömfors,Jonesjö-81] O. Strömfors, *The implementation and experiences of a structure-oriented text editor*, SIGPLAN/SIGOA Symposium on Text Manipulation, Portland, Oregon, June 8-10, 1981.

[Teitelbaum,Reps-81] T. Teitelbaum, T. Reps, *The Cornell Program Synthesizer: A Syntax-Directed Programming Environment*, CACM 24:9, Sept 1981.

[Teitelman-78] W. Teitelman, *Interlisp Reference Manual.*, Xerox Palo Alto Research Center, 1978.

[Uhl,et.al-82] J. Uhl, S. Drossopoulou, G. Persch, G. Goos, M. Dausmann, G. Winterstein and W. Kirchgässner, *An Attribute Grammar for the Semantic Analysis of ADA*, Lecture Notes in Computer Science, Vol 139, Springer-Verlag 1982.

[Wilander-80] J. Wilander, *An Interactive Programming System for PASCAL*, BIT 20:2, 163-174, 1980.

2.3 Other PELAB Projects

2.3.1 Tools for interactive program development.

Dan Strömberg's licentiate thesis is near its completion. It is on *Tools for Interactive Program Development* and contains a number of contributions from the period 1980 to 1984. Part of his work has been somewhat DICE related.

2.3.2 Previous Thesis Projects

Over the years of the running STU program, PhD theses have also been presented by the (former) PELAB members Pär Emanuelson 1980, Jan Komorowski 1981, Östen Oskarsson 1982, and Hans Lunell 1983. They started their work in the seventies, before PELAB was founded. Their contributions have been presented in earlier progress reports and will not be elaborated further here. However, their work together with results from Kenth Ericson, Jerker Wilander, and others, have made up the template upon which current PELAB activities are built.

2.4 External Contacts

The PELAB policy is to send new students to international summer schools and workshops on relevant topics. Later, when they have results to present, they are encouraged to attend recognized conferences.

- *Rober Bilos* has attended a two day course on Validation and Verification in Göteborg, and a two week summer course on Compiler Construction and on Code Optimization and Code Generation at Stanford University.
- *Johnny Eckerland* has been to a workshop on System Development Environments, Aarhus University, and attended a one week conference on Advanced Personal Work Stations on Capri, 1984.
- *Pär Emanuelson* has spent three weeks at Computer Science Department, Leiden University. He has also been to a workshop on System Development Environment, DIKU, University of Copenhagen, and attended a one week conference on Advanced Personal Work Stations on Capri.
- *Kristina Ernstsson* has attended a seminar on Interactive Development Environments in Stockholm, 1982, a workshop on Program Development Tools, Lund, 1983, a seminar on Expert Systems in Stockholm, 1983, and also a one week conference on Advanced Personal Work Stations on Capri, 1984.
- *Peter Fritzson* has visited or given talks at CMU, Xerox PARC, Sun Micro Systems, Aarhus University, Cornell University, and Harvard University. He has also presented papers at ACM SIGSOFT/SIGPLAN Software Engineering Symposium on High Level Debugging, Pacific Grove, March 1983, and at the ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments, Pittsburgh, April 1984.
- *Anders Haraldsson* has visited AAAI-84, ACM Symposium on LISP and Functional Programming 1984, Stanford University, MIT, and Symbolics
- *Mariam Kamkar* has attended a workshop at DIKU, University of Copenhagen on Static Analysis of Programs. She has also taken a one week summer course on Code Optimization and Code Generation at Stanford University, and attended FCT 1983.
- *Bengt Lennartsson* is member of Ada Europe Environment Working

Group, follows Ada environment activities in general, has been to workshops on System Development Environments at DIKU, University of Copenhagen, and DAIMI, Aarhus University.

- *Nahid Shahmehri* has attended a workshop at DIKU, University of Copenhagen, on Static Analysis of Programs. She has also been to a one week summer course on Code Optimization and Code Generation at Stanford University, and attended FCT 1983.

Visitors to Ida are listed on the institute level. Among visitors of particular interest for PELAB are Warren Teitelman, Pierre Deransart, and Tim Teitelbaum.

PELAB has also been visited by representatives from the companies: Ericsson Information Systems, Ericsson Radio Systems, LM Ericsson, Philips Elektronikindustrier, Telelogic, Televerket, Asea, and others.

Anders Haraldsson is member of a group developing a program for continued education for a group of Swedish industries, the Oktagon group. He has also been responsible for a course *Structure and Interpretation of Programs* for programmers at LM Ericsson and Ericsson Information Systems.

In the Ida series of seminars for industry, the SOFT seminars, PELAB has been involved in SOFT-2 on Software Development Environments and SOFT-3 on Logic Programming. An industry oriented workshop on *Program Data Bases* was arranged in November, 1984.

PELAB activities have also been presented on many other occasions at various meetings, seminars, *etc.* in Sweden.

PELAB will also will be involved in the Knowledge Transfer Program, KTP, presented in an appendix.

2.5 Some PELAB-related Master Thesis Works

Some of the undergraduate master thesis works have been more or less related to the research in PELAB.

Per-Ove Jakobsson: *Test Aid PROB.*

Roland Rehmnert: *A Syntax Analyzer for the AXE Command Language.*

Stefan Frennemo: *Automatic Generation of Scanners.*

Reibert Olsson: *An Ada Parser and Pretty-Printer.*

Hans-Göran Puke: *Interface to Operating System for a Business Oriented Program Package*

Pernilla Nordström: *Response Time Measurements on the Wilbur System with the aid of an IBM PC.*

Per Öqvist: *Error frequency versus Size and Complexity of Program Modules.*

Mats Winberg, Bengt Rosen: *Demonstration of the Program*

Development System PUS 80.

Anders Svensson: *Benchmark of the Real Time Facilities of ERIPASCAL and ERIOS.*

Rober Bilos: *Syntactic Error Diagnosis and Recovery.*

Maria Marklund: *Investigation of the Performance of the Operating System CP/M-86 for the Development and Maintenance of Hardware Test Programs.*

Kerstin Olsson: *A Language Transformer - LISP to PASCAL.*

2.6 Future Direction

We have established knowledge and experience in our area, and the ambition is to continue along the present path. A kernel project is intended to live for at least one generation of graduate students. After the development of some missing tools and the database in DICE, it would be natural to consider distributed targets. The new profile, Programming Support Environments for Distributed Targets, is highly relevant for Swedish industry, and it rises a number of scientifically challenging problems. We have also a very suitable background to attack the area.

Among the specific problems to study are:

* how to extend the powerful incremental architecture developed in the DICE project to handle the much more complex situation with distributed targets ?
* how to monitor, debug, and test concurrent programs in general ?
* how to design an appropriate user interface for the communication with an executing concurrent system ?
* how should the communication links between the host and the distributed target be designed and used ?
* how to compose the tool set supporting the code management for a distributed target possibly consisting of different types of processors ?
* investigate mechanisms for dynamic allocation of processors for processes in an executing program

The goal is not to give significant contributions to every problem listed above, but rather to focus on the architecture of the supporting host system and on its communication with the target and the target processes. The research is intended to be based upon existing modern hardware (powerful work stations for the host, local area networks for the links, *etc.*). We will also use existing programming languages and notations, for example, CCS, Edison, PASCAL, or Ada.

In the five to ten year perspective, however, we plan to direct kernel projects towards the next generation of languages. It would be quite possible to apply the basic philosophy, that of saving information and dependencies and use incremental algorithms, also for incremental compilation of, for instance, rule based systems.

3.

ASLAB
The Application Systems Laboratory

The research program in the Applications Systems Laboratory (ASLAB) is oriented towards the development of more efficient methods for developing and using computer support in special application areas. That is, assuming that we restrict ourselves to a specific domain or a particular class of problems, how can we support software development, maintenance and utilization more efficiently than by using e.g. general-purpose programming languages and a conventional staged development process with strict phases.

Within the laboratory there is also a tradition of work on office systems, including logic-based formalisms for description of information structures and incremental operations upon these structures. Recently these studies have been broadend into monitoring and planning in real-world systems and control of reasoning. Those issues will from now on be studied within a new group, *The Laboratory for Representation of Knowledge with Logic*, which will be led by Erik Sandewall and formed by a split-off of the current Office Systems Group within ASLAB and Jim Goodwin from the AILAB.

3.1 Researchers and projects

The main current achievements in the laboratory are:

1. The coordinated development of office information systems software and a novel theory for information management systems (IMS), where piecewise operations on network information structures are the most important ones to be described. A tutorial material on this theory has been written by Sandewall and is currently made the basis of projects were its proper use is investigated (Sandewall, Rönnquist). A prototype software system (IM4) and hardware (M68000-based high resolution display and laser printer) has been developed.
2. The forming of a project for investigation of how knowledge-based techniques (expert systems) can be used together with more conventional approaches in application development (the K-Base project, Hägglund et al.). This project has been preceded by two expert-system implementation projects in real-world settings (medicine and industry), which serve as an empirical basis for continued efforts.

3.1.1 Personnel, Aslab 1983-84, 84-85

The following list presents persons active in Aslab during the previous and the present year.

	% 84/85	%83/84
Laboratory leadership:		
Sture Hägglund, PhD	70	60
Britt-Marie Nygren, secr.	25	25
Project leadership/thesis supervision:		
Christian (Gustafsson) Krysander, lecturer	50	50
Erik Sandewall, PhD, professor	30	25
Bo Sundgren, PhD, adj professor	20	20
Employed graduate students:		
Johan Andersson	25	--
Mats Andersson	--	50
Shamsul Chowdhury	50	50
Arne Fäldt	50	100
Henrik Nordin	30+50	--
Roland Rehmnert	30+50	50
Ralph Rönnqvist	30+50	50
Kristian Sandahl	30+50	50
Ola Strömfors	--	25

(*Comment: Time which is not accounted for above is usually used for teaching. When two figures are given, the first relates to duties within the knowledge transfer program.*)

Technical services:		
Jan Axing, research engineer	--	50
Leif Finmo, research engineer	--	50
Sven Moen, research assistant	25	--
Stefan Wrammerfors, research assistant	75	--

Guest researcher:

Dimiter Driankov (Decision support systems and fuzzy reasoning)

Associated personnel:

Lars Bengtsson, S-E-Banken AB
Jan-Olof Brüer, Dept of electrical engineering
Christian Forsäng, Volvo-Data AB
Hans Gill, Dept of medical informatics
Torgny Pettersson, Ericsson Information Systems AB
Bengt Rosén, Ericsson Information Systems AB
Toomas Timpka, Dept of medical informatics

Anders Wallgren, Dept of math/statistics
Britt Wallgren, Dept of math/statistics

3.1.2 Summary of current activities and results.

During the last year a strong effort has been made to fulfil the expectations from STU that there will be an increased application emphasis and knowledge transfer to industry during the final two years of the 5-year knowledge development program. Thus we e.g. accepted to carry out a commission for ASEA engaging 2-3 persons during the spring. We have also taken a large part of the responsibly for the development of the Industrial Knowledge Transfer Program (which is described elsewhere in this report).

This period has also coincided with a generation shift in our personnel. Thus a group of new graduate students has been admitted and are now working through the initial phase of their studies with an emphasis on the course part. We feel that participating in the cooperation with industry personnel in the knowledge transfer projects is a relevant and rewarding activity, which we use to reduce the teaching load on new graduate students.

The main activities in the laboratory during 1983-84 has been a further development of the theory for information management systems and work on so called knowledge-based systems and their integration in practical application environments.

From the administrative point of view, we are working in two project groups, one for Office Systems (*Sandewall*) dealing with special tools for interactive information processing and the theory for their description, and one for Database Technology and Application Development (*Hägglund*) dealing with database interfaces and the integration of expert systems into mainstream software technology.

During 1983-84 Bo Sundgren of the Central Bureau of Statistics, Stockholm, formerly part-time professor at Uppsala University, accepted a 20% part time professorship ("adjungerad professor"). His tasks are mainly to broaden our competence in the field of database technology and to build up a group working with statistical information systems.

3.1.3 Background.

Work is the laboratory is oriented towards research on specialized tools for development of applications software, with an emphasis on personal information management systems and office systems. Methodology and tool systems are developed in close cooperation with external parties in joint projects. Cooperation with the other laboratories on fundamental research issues is also emphasized.

The long-range goal of the work is to make computers easier to use and thus to promote the development of more powerful computer-based support systems for various tasks. For this purpose we expect results from AI research to be an important source of inspiration and thus that knowledge-based expert systems

and especially their relationship to more established areas of computer science is a prime target for projects in the laboratory. Our approach is based upon the belief that we have to advance our understanding of how integrated interactive software environments supporting more problem-oriented concepts can be realized in an efficient fashion.

Two facets of this problem are studied in our laboratory:

- One is the design of interactive computing environments for professional users, i.e. users who are experts in their own application domain, but not necessarily proficient in computer-oriented aspects of programming.

- The other is the development of higher-level languages and supporting tools for construction and maintenance of software, i.e. techniques for producing programs expressed in a less procedural and more application-oriented terminology than todays general-purpose languages.

The 1980 application to STU when ASLAB was founded identified the following technical areas which should be approached in order to simplify access to computerized services:

1. *Modelling of applications, to be studied with respect to how the applications are described to the computer, and how the processing within the computer is organized. Specialized tools for this purpose may be:*
   - *specialized programming languages.*
   - *program generators, which accept a description of an application and generate a corresponding, tailor-made program.*
   - *general-purpose programs ('superroutines') which are highly parameterized, and which enable the user to develop an application by selecting appropriate parameter structures.*

2. *Dialogue techniques, to be studied with respect to the appearance of the dialogue, the syntax of the dialogue data and the pragmatics of the relationship between the user and the system.*

3. *Development of very-easy-to-use systems for specific purposes.* (Since this task is rather a question of product development than build-up of technological knowledge, the research proposal was restricted to the first two areas.)

Today there is a rapidly growing interest in theses matters within many application areas, and we think the motivation for research efforts are even stronger now than five years ago. This development is paralleled by the insight than future solutions might be sought in the direction of *knowledge-based* systems. We quote from the previous proposal:

> *Expert problem-solving systems in artificial intelligence research are an early example of that possibility (* ... to keep knowledge about an application present at all times ... *), although we believe that it is more fruitful to embed AI techniques in current computer applications, rather than inventing new applications. Building systems which contain and use an explicit description of their application environment, is an*

approach to ultimately obtain systems which

- *are able to explain to the user, the reasons for their behaviour or response in specific situations;*

- *allow the user to request (in the language of the application) exceptions from the standard procedure usually performed by the system, and have it preformed correctly and consistently;*

- *are able to handle 'new' situations adequately, and request the user's help only when really needed.*

This expresses a view of software architecture which still is central for research activities in ASLAB.

3.2 Office systems

(*Sandewall et al.*)

The work in the office systems group has the following four goals:
- development of models for "office procedures", or things that happen in offices;
- development of a theory that (in an empirical sense) accounts for observed phenomena in office software;
- development of prototype office workstation(s) with interesting and significantly new properties;
- development of a cluster of workstations with a resolution that supports standard facsimile representation of A4 size documents, for use as a tool by other projects.

The members of the group are/were: Erik Sandewall, Ralph Rönnquist, Mats S. Andersson, Arne Fäldt, Johan Andersson, Jan Axing, and Leif Finmo.

During the reporting period October 1983 - September 1984 there has been activities on all four goals, but in the following different ways.

3.2.1 Office procedures.

The results a few years ago on information flow models are now continued by the Department of Medical Informatics, in cooperation with Ericsson Information Systems. We maintain a working contact with the activities there.

In the continued work, we wish to go towards more expressive modelling methods, which in particular should make it possible to express a broader view of the office work, and not only the information handling aspect as in the information flow model. This interest is shared with the information theory group in our EE department. They study security issues in office systems, and need an office model for that purpose. During the winter 1983/84, a graduate course on modelling of office procedures was organized jointly by our two

groups, and led by Jan-Olof Brüer from EE. The course resulted in an Aslab memo which reviews a number of published results in the area (ASLAB Memo 84-01).

The study of existing literature did not however suggest any good topic for our own continued work. We now turn to the A.I. literature on knowledge representation for planning and problem solving, and especially the non-standard logics which are being tried out there (various modal logics, probabilistic logic, etc). What we are looking for is, in short, a logic for the description of sequences or other structures of actions. It is assumed, then, that those events have a duration in time, and may proceed in parallel. Our graduate course on "A.I., expert systems, and representation of knowledge" during the fall term of 1984 concentrates on these topics.

The description of event structures is highly relevant for a number of other purposes besides office systems. Some of them are already being studied in cooperation with a few major corporations.

3.2.2 Theories for office software.

The effort during the reported period has concentrated on our emerging *theory of information management systems*, or IMS theory for short. The key idea in this theory is to view a data structure or data base as a simple binary network, with nodes, arcs from nodes to nodes, and arcs from nodes to attribute values. Such networks are viewed as interpretations for propositions in a variety of three-valued first order logic (with undefined as a truth-value because interpretations may be partial). At the same time, algebraic operations are defined for composing networks into larger networks. There is of course an analogy with the propositional and algebraic ways of treating relational data bases.

The main framework as well as the rationale for it were outlined in a set of lecture notes (LiTH-IDA-R-83-03). A particular result regarding how derived knowledge should be viewed, and non- monotonic inference, were also reported separately (LiTH-IDA-R-83-01).

Ralph Rönnquist and Erik Sandewall, in a joint work, have showed how the propositional IMS theory can be used for characterizing the elementary data structures of ordered trees (represented as linked lists a la Lisp) and unordered trees. Two different ways of defining the relationship between the two trees were found, and it was shown that one of the definitions was stronger than the other. The "bottom line" is that IMS theory can be a useful tool for characterizing data structures built from pointers.(LiTH-IDA-R-84-04).

Presently, Ralph Rönnquist studies how relational data bases (including the schema information) and the algebraic operations on relations, can be represented in terms of IMS theory.

3.2.3 Prototype office system.

During the reported year, work has been pursued on several parts of the intended system, but some aspects of the work have been delayed because Erik Sandewall was on sabbatical during the spring of 1984, and Johan Andersson was in military service from June, 1983 to August, 1984.

Our intended system will consist of a controlling system which directs a number of lower level processes, on several processors. The controlling system should be a Lisp implementation of IMS theory, and has the working name IM4. The present implementation of IM4 runs on DEC-20. The subordinate processes include both some that run on the DEC-20 (especially for text formatting and computer mail), and some that run on dedicated M68000 systems (especially for bit-map maintenance and other graphics).

The work during the year includes:

- completion of an M68000 based driver system for a Canon laser printer (Axing, Finmo, Fäldt, Ström)

- completion of an M68000 based driver system for a high resolution display (Finmo, Fäldt)

- work on text formatting software which makes it possible to format text on the DEC-20 and send it to the above devices for presentation. Some parts of the TEX system (including its fonts, the metafont package, etc) have been used (Fäldt);

- interface procedures which make it possible to invoke various services (mail, text editor on small text segments, display control, etc.) from the Lisp programming system (J.Andersson, J Axing)

3.2.4 Facsimile level workstation

During previous years a 68000 based driver system for a Canon laser printer was built. During the year 1983/84 a similar system driving a high resolution display was also built. It was decided not to go directly to facsimile level resolution in the display, but to first go to a more moderate level.

Meanwhile, we have been watching the market for newer equipment that could make the full facsimile level resolution available in a screen at moderate cost.

3.3 Database Technology and Application Development

3.3.1 Knowledge-based application development.

(*Hägglund, Rehmnert, Sandahl, Nordin*)

This work is oriented towards the application of knowledge-based techniques for software development, in particular the integration of methodology for developing expert systems with more conventional information technology such as database management and office information systems. In this process we emphasize the potential benefits of AI methods for producing more useful, easy-to-change and understandable software, rather than as a way to solve computationally hard problems.

The current project group consists of graduate students who now have passed their first year. Work up till now has involved participation in applied projects (see below) in addition to taking courses.

A3 - Antibody Analysis Advisor

Last year a simple medical expert system was developed for the purpose of providing guidance in the initial selection of analysis techniques for antibody identification in blood samples. EMYCIN, a tool for implementing expert systems, was used as the vehicle for gaining experience of the knowledge acquisition process. Evaluation of the project showed that the A3 System produced more reliable recommendations than those actually carried out in a historical comparison and that a 10% increase in efficiency (eliminating uninformative tests) could be expected.

Using the system on a routine basis would however presume a migration from the DECsystem-10 Interlisp-based implementation to the existing laboratory systems on a Vax computer. Then substantial parts of the interactive data entry process could also be substituted by accesses to computer-stored patient data.

We feel that such a migration process will be typical in many applications where the expert-system approach is tried. That is, the expert system is developed as aid to understand the domain and the problem to be solved, to work out the delimitation of the task, to promote the formulation of relevant knowledge and to tune the system until a certain level of competence is accomplished. Then in many cases it will turn out that not the full power and flexibility of the development tool is needed for the production system, and a reimplementation or migration will be undertaken. For instance, we have exactly this experience also from the previous MEDICS project, where tutoring programs to be run on a PC are generated from an expert-system-like development environment.

GARMAN - Sales Support for Spot-Welding Robot Configuration.

This project was carried out as a commission from ASEA with the main purpose to demonstrate the feasibility and business opportunities of expert

systems for ASEA. From our point of view it served as a training project for two graduate students (Rehmnert and Sandahl) as well as a test of the possibility to reproduce results from a more research-oriented setting. The system was delivered on time and was a complete success as a demonstration of what could be accomplished.

The main task of the system is to support a sales engineer in the configuration of a spot-welding robot. Important aspects are to secure that relevant and complete information is acquired from the customer to suggest a suitable combination of equipment and to verify that electrical and mechanical constraints are satisfied. EMYCIN was used as the basic tool although the dynamic nature of the problem (initial assumptions may have to be relaxed) resulted in a solution where significant parts of the problem were solved directly in Lisp.

Other explorative implementations

As a part of the learning of the new Interlisp-D programming environment on the Xerox 1108, we have undertaken some minor implementation efforts, motivated primarily be the desire to explore what the graphics interface can add to previous approaches. Thus we have investigated how the information flow models previously developed for office information systems can be supported in a CAD-like style on the 1108 work station.

Further we have implemented a version of the simple "expert system" tool Expert Ease for developing classification models based on Quinlan's ID3 algorithm. The system takes advantage of Interlisp-D graphics facilities and presents the user with a particularly nice interface for developing and running consultations.

Plans for 1984-85: The K-BASE project.

The main purpose of the K-BASE (Knowledge-Based Applications Software Environments) project is to study how components of expert systems and the related methodology of knowledge acquisition and implementation can be integrated with more conventional approaches to development of applications software. To be more precise, we are interested in expert systems not as a way to solve hard problems but as a way to introduce the following qualities in the software development, maintenance and use:

1. Interactive support for application modelling, through the use of e.g. rules as a way to incrementally assert information to the system.
2. Advanced dialogue management, including semi-natural language explanations and queries.
3. Learning support, based on the fact that information inside the system may be inspectable and also reusable for teaching purposes.
4. More maintainable systems, since the distance between what is stated by the domain expert and what is entered into the system is shorter than in conventional programming.
5. Less rigid tools for application development than today's fourth generation languages which concentrate on superficial similarities between tasks (forms management, report generation, etc.)

We assume an architecture where the final system should run not only inside the development system but also on e.g. a personal computer or in a corporate database environment. This implies that we have to study how a core system can be realized in different environments and how the application dependent part can be "compiled" or migrated (e.g. through a manually supervised transformation process) to a target system. Several alternatives should be contemplated for the migration process in addition to moving to an equivalent (Lisp) system, including such possibilities as generating compilable programs in a suitable language or some kind of decision tables to be interpreted. The purpose of migration may be:

- to promote target system independence,
- to interface to existing software,
- to improve economy,
- to increase efficiency,
- to hide development features, or
- to make the system more robust.

We will approach this problem by designing a development environment for the Interlisp-D work station, possibly using some existing software (e.g Loops) as the core of the system. Each component in this development system will be designed under the constraint that a counterpart, when appropriate, should be possible to realize in the target environment. Part of the project is to define reasonable target environments, with interfacing and integration with database management as central topics.

In order to further limit the task and make the problem area manageable, we intend to restrict ourselves to consultation system, i.e. systems providing advice or decision support. In particular we will study initial-advice systems, where the system supports a non-expert user to handle routine or almost-routine cases while more exceptional cases are to be forwarded to a human expert. This basic pattern reappears in several projects, where we are engaged in external cooperation, which makes it very promising as a study object. Our applications are:

a) sales support in process industry,
b) financial and legal advice in banking and
c) medical treatment in primary care.

Subprojects (for licentiate theses) will be:

1. Investigation of core functions in knowledge-based systems, such as knowledge representation schemes and inference engines. The purpose is to identify a useful subset and demonstrate how they can be implemented in the development and the target environments.
2. Development of a migration methodology, stressing the view of the development environment as a tool for knowledge acquisition. Work involves a classification of possible strategies for migration and the selection of mechanisms for transforming and downloading to the target environment.
3. Study of reusability tools in a knowledge-based application development environment. Assuming that we want to customize applications from a common base, it is only natural to think of "meta rules" encoding expertise supporting the selection and adaption of

predefined building blocks.

In addition the following issues are candidates for treatment:

4. Incorporation of expert system features in a relational database query environment.
5. The possibility to make explanations and teaching facilities available in the application environment on a semi-automated basis.

3.3.2 Technical Information Systems.

(*Gustafsson-Krysander, Andersson*)

This project is a cooperation with the department for engineering materials, which carries out the main part of the work. Research is concentrated on the subject of databases for engineering materials. The task is to support the selection of materials in such a way that consideration of new materials, e.g. composites, is encouraged. Thus human-computer communication and cooperation are important aspects of the work, which also involves the study of work station-based graphics facilities.

3.3.3 Statistical Information Systems.

(*Sundgren, Chowdhury, et al.*)

This group was started during 1983-84 when we Bo Sundgren joined the department as a part time visiting professor ("adjungerad professor", 20%), with partial funding from the National Bureau of Statistics, Stockholm. This group will however keep a close contact with the other ASLAB projects working with theory for information management systems, database environments and expert systems.

The group has identified the following main issues for potential studies:

1. **Utilizing administrative data for statistics production.** The idea is to study how existing data can be used also for statistical purposes, without loss of quality.
2. **Metods and tools for automated production of statistics software.** This area can be seen as a specialization of query languages and program generation techniques for databases in general.
3. **Human-computer interaction in statistics production.**
4. **Management of uncertain and incomplete data.** Especially the relation between quality measures in statistics production and uncertainty as handled in expert systems will be studied.
5. **Systemeering methodology for statistical information systems.** This area continues Bo Sundgrens earlier work on infological and conceptual modelling.
6. **Medical applications of statistical information systems.**

These areas were selected since they represent both areas where competence and/or cooperative activities are possible as well as areas where significant research problems can be identified. Initially a few of them will be selected. Work in area 1 has recently started in cooperation with the department for mathematics/statistics, and work in area 6 is being planned together with the department for medical informatics.

Bo Sundgren developed a course and lecture notes on *Conceptual Design of Databases and Information Systems*, which was given also at Astra in Södertälje. A series of seminars was held in the spring resulting in a report on the research program on statistical information systems (ASLAB Memo 84-02).

3.4 External cooperation.

Aslab projects emphasize joint efforts with other groups and industry. The following are the main current involvements:

1. Department of Medical Informatics. Continued cooperation on information flow models and rapid prototyping techniques (Gill). Currently the information flow model is reimplemented as a product as a commission from industry. Previous cooperation on advanced CAI systems (MEDICS) is now followed by joint work on medical expert systems, where a physician has enrolled as a PhD student in medical informatics (Timpka).

2. Department of Electrical Engineering, Information Theory group. Cooperation in the area of office information systems, in particular models of office activities (Brüer).

3. Department for Construction and Production Technology concerning the design of database systems for selection engineering materials. (See technical information systems above.)

4. ASEA. Cooperation in the area of expert systems. Joint projects are pursued and participation in the Knowledge Transfer Program is being planned. (Törne.)

5. Ericsson Information Systems and S-E-banken. Joint activities in the Knowledge Transfer Program with an emphasis on intelligent knowledge-based consultation systems in business and banking and on other end-user systems. (Pettersson, Rosén, Bengtsson).

6. National Bureau of Statistics. Study of the design of statistical information systems. (Sundgren et al., see above.)

7. Nordic cooperation with Oslo (Kristen Nygaard) and Aarhus (Lars Mathiassen) in the SYDPOL programme (System Development Environments for Profession-Oriented Languages.) This programme is partly supported by Nordforsk and consists of national projects and four inter-nordic working groups, where Aslab participates in those concerned with systems developments methods and human-computer interaction techniques.

3.5 ASLAB Memo series 1984.

In addition to external publications and departmental reports, ASLAB has instituted a Memo series containing working papers etc. The following reports has been issued or are planned:

84-01 **Brüer, Chowdhury, Fäldt, Gill and Rönnquist:** Office Models.

84-02 **Sundgren, et al.,**: Statistiska Informationssystem (Statistical Information Systems).

84-03 **Rönnquist**: Customizing Command Languages Dialogues with the YAKI package.

84-04 **Moen**: En implementering av Expert Ease under Interlisp-D.

84-05 **Rehmnert, Sandahl**: Implementing an Expert System for Spot-Welding Robot Configuration.

84-06 **Hägglund**: Introduktion till kunskapsbaserade expertsystem i ekonomiskt-juridiskt arbete.

84-07 **Hägglund, Nordin, Rehmnert and Sandahl**: Towards Knowledge-Based Applications Software Environments. Project Proposal.

84-08 **Nordin,** On the Use of Meta-Level Knowledge to Support Reusability in Knowledge-Based Systems.

4.

AILAB
The Artificial Intelligence Laboratory

4.1 Researchers and Projects.

From July, 1st, the AI laboratory is directed by Erik Tengvald. Uwe Hein, former leader of the laboratory, has left the University and started a spinn-off company, EPITEC AB, together with some other persons from our department. Consequently, the AI laboratory is currently being reshaped and some new directions for research will be taken. Roughly speaking the activities since the start of the laboratory in 1981 until now can be divided into three phases:

1. Start up and definition of the scope and focus for research. During this period intense recruiting took place, many internal seminars were run and the scope and focus for research were discussed and decided. At that time research in the laboratory was committed to knowledge representation, reasoning and natural language. Also expert systems methodology was chosen as a vehicle for practical experimentation.

2. After the start up time, there naturally followed a, still lasting, period of concrete research projects. This period includes the thesis works by E. Tengvald and J. Goodwin, as well as the work on the object-oriented representation system PAUL.

3. The third period also follows naturally after period 2. Experiences from research in the laboratory, particularly on PAUL, but also the research on expert systems in general have now led to commercial applications at EPITEC AB, where both Hein and Lund are engaged.

The activities within the reorganized AI laboratory will from now on be concentrated around a kernel project. This project is planned to concentrate on development of software and theory for ***knowledge representation system writing systems***, emphasizing e.g. declarative approaches and constraint-based formalisms.

4.1.1 Laboratory Members

	% 84/85	% 83/84
Laboratory leadership:		
Erik Tengvald, PhD (1984/85)	100	--
Uwe Hein, PhD (1983/84)	--	90
Lisbeth Linge, sekr	35	45
Employed graduate students:		
Sten Erik Bergner	--	50
Nils Dahlbäck	80	--
Jim Goodwin	50	100
Arne Jönsson	20	--
Tomas Lund	--	75
Jalal Maleki	75	50
Ludmila Ohlsson	--	75
Erik Tengvald	--	100
Mats Wirén	80	75
Technical services:		
Bernt Nilsson	50	50

Associated personnel:

Hans Grunditz, Dept. of Production Engineering (1983/84)
Anders Törne, ASEA (1984/85)

4.2
AI Programming Systems; Mechanical Engineering Applications

The mainstream of activities in the AILAB has focused on the design of programming systems for A.I., as studied through applications selected from Mechanical Engineering. The programming systems concepts have particularly been object-oriented programming and lately programming through constraints. The major application area that we have studied, namely production planning for turning, involves some incompletely understood problems such as the handling of geometrical information.

One of the experiences of this project is that the mainly procedural object-oriented programming systems concepts is insufficient for handling geometrical problems. Currently we believe that programming through constraints is better suited for such problems, mainly because of its more declarative nature.

4.2.1 PAUL

During the autumn 1983 minor additions were made to the object-oriented PAUL knowledge representation system and experiments have been performed with the design of graphics based interactive tools. Although research on PAUL has been concluded, experience from the project will have serious impacts on a commercial project at EPITEC AB to develop knowledge engineering tools.

4.2.2 OBS

In the OBS project we attempted to build a system able to construct operation plans for turning of parts in the machining industry.

4.2.2.1 The problem

Such a system starts with a problem formulation composed of three parts:

* Processing equipment descriptions. More precisely descriptions of the lathe, the available clamping tools and the available cutting tools.

* A specification, in our case a drawing of the blank. Including material information.

* A drawing describing the finished part. Besides being a geometric description of the part it contains tolerance information of different kinds.

Output from the system should in principle be a NC-part program. However it is sufficient with a sequence of operations described with enough precision to work as input to current CAM software.

The planning is performed under the following two most abstract correctness constraints: The plan should not harm the equipment and the machined part should satisfy the specified tolerances.

Furthermore one plan is better than another if it entails lower cost. The planner should at least sub optimize with regard to the cost constraint. An alternative to the low cost constraint is the low time constraint. We did not consider this alternative in our work.

4.2.2.2 Overall description of our systems mode of work.

Our system have to some extent been built to mimic the behavior of a human operations planner. There are however some differences.

When observing a human planner one can observe two parts of the planning process. We have an intuitive part where the planner performs geometrical reasoning and decides on the overall structure of the plan. This is followed by a more "formalized" part where the planner, using different kinds of tool

descriptions, material descriptions and such like, decides on the specific values for the plan parameters, a kind of technological reasoning.

For a human planner these steps are clearly discernible. In fact the intuitive part usually only takes minutes, while the technological part can take days.

In our system the situation is reversed. The intuitive geometrical reasoning is complex and takes time. Finding the geometry of the suboperations is especially cumbersome since it entails a lot of search. The technological reasoning on the other hand is simple because of its "formalized" nature. Consequently we merge the two phases so that technological constraint can prune early. This reduces the overall need of backtracking.

On a concrete level our system works by finding the basic operations, finding important features of the object as grooves and rings, refining this classification using information about the actual tool set available, deciding on where to have finish cuts and the finish cutting margins, deciding on suboperations and tool selections, selection of machining parameters.

4.2.2.3 Technical, organizational and some scientific experiences.

The project was too big. As a consequence we have not been able to complete the system into a running whole, the address space of the DEC-20 is not enough for our code.

The project organization for a project of this size have to be more goal directed than the one usually used by university researchers. It is one experiment, not many.

Geometrical reasoning is very hard. For example our fairly simple search schemes, used for finding suboperations, would probably break down when confronted with parts of real world complexity.

Because of its procedural style object oriented programming is uneconomical for solving more complex problems. One has to supplement with more declarative inference engines as constraints or rules.

The two last points was stressed by the experience of the KLAMP subproject [Grunditz83]. In this subproject part of the geometric reasoning, selecting clamping tools, were attempted using an object oriented formalism.

4.2.2.4 Major scientific experience.

In a more general sense the main result of the OBS work is the following: The main concern of practical AI work is to suppress design complexity. Time complexity is disregarded as long as it does not become too disturbing. Consequently the central concept for AI theory is design complexity.

These ideas have been further developed in [Tengvald84a].

4.3 Reason Maintenance Systems.

During the last years Jim Goodwin has been working with reason maintenance systems. Now when we redirect the AILAB towards constraint inspired formalisms his basic theoretical work will come to good use.

We believe that reason maintenance systems will have an important role in the future of AI. Reason maintenance is important in all systems, where any kind of structures has to be revised. A pertinent example is the world model of an intelligent robot. Reason maintenance ideas have already come to practical use in the more advanced Visicalc style systems currently available in the marketplace. Moreover it is not impossible that the non-monotonic logic corresponding to reason maintenance systems, can become an important part of a future mathematics of knowledge representation.

4.3.1 WATSON project (Jim Goodwin)

Jim Goodwin's WATSON project is concerned with one of the most central topics in Artificial Intelligence: computer reasoning. The classical tool for studying reasoning is formal logic, but traditional logics do not meet the needs of Artificial Intelligence for a theory of reasoning in finite, physically realizable reasoning agents, such as people or computers.

Real agents must often reach decisions despite inadequate information and under time pressure. To complement their limited information, they make assumptions about their problem domain. While they continue thinking about the problem, they must constantly choose to investigate some lines of reasoning, and ignore others which are equally valid but appear less relevant or promising. And to reach a decision at all, they must eventually decide to stop reasoning and act on the basis of their analysis so far.

Rational agents certainly do not make such choices randomly. We hold one another accountable for the reasonableness of such choices. If you returned a book to the reading room five minutes ago and I ask you where it is, I would feel you lied if you said you did not know: you are obliged to assume it is where you left it, even though somebody else might have taken it already. Yet if you actually saw Sally take it, you would be obliged *not* to assume that it was still there, even though she might by now have put it back. Classical logics have nothing useful to say about when such assumptions are rationally chosen and when not.

Recent work in non-monotonic logics has attempted to deal with the problem of how assumptions about the problem may be reasonably introduced. Unfortunately, the solutions achieved still fail to deal with the need to act on incomplete reasoning due to time pressure. The resulting systems are in several cases formally undecidable, which makes them practically useless as criteria of rationality in real systems.

The recently developed programming technology of *dependency nets* and *reason maintenance* provides an alternative, however. Systems have been built which can make assumptions, and then later abandon them if need be and recover to a consistent state. The "change of mind" may result because new *a*

priori information is discovered, or simply because the system thought further about information it already had and discovered that the assumption was not tenable.

However, such systems have not been presented clearly as "logic programming languages" or as implementations of any other clear formal idea. The reason is the lack of any logic or clear formal idea which they might be implementations of. This has also held back development of the systems' actual capabilities. A clear formal model is therefore sorely needed.

Of course AI has been in this dilemma for many years; the debate about the suitability of logic for Artificial Intelligence's purposes is of long standing. The difference is now that the dependency-based implementations provide a clear suggestion of a new formal framework which might serve the purpose better than traditional logics (including non-monotonic ones).

Our efforts have produced a formal theory called Logical Process Theory and an implementation called WATSON (written in Interlisp). The theory defines, for any given state of the reasoner, what things it must believe. This includes not only the conclusions which are proven so far, but also the set of assumptions which the agent must reasonably make in this state. Because assumptions may be dictated in one state and yet ruled out in a later state, the set of things believed may vary non-monotonically. Such "changes of mind" may occur either due to the provision of new *a priori* information or simply to thinking further about information already available. "Non-monotonic logics" cannot represent the latter case, since they have no representation of the process of reasoning.

Finally, the implementation of WATSON has gone a step further, to provide a model implementation of one form of *reasoned control of reasoning.* This means that whenever a choice is made to follow one line of reasoning and ignore another, the machine can defend that choice by a currently valid *control argument.* Control arguments reason from the machine's current beliefs about the world and the problem domain, using rules (including domain dependent rules) about what sorts of things are useful to think about in what sorts of situations. The conclusion of a control argument is always that such-and-such a line of inference is still worth pursuing further. The extension of the formal theory to include reasoned control is a topic for future research (after the upcoming dissertation on WATSON is presented).

Jim Goodwin is currently finishing a doctoral dissertation within AILAB titled "WATSON: A Dependency directed inference system". The work presents a theory and implementation for non-monotonic reasoning, and a technique for controlling inference called reasoned control of reasoning. Both are based on the technique of dependencies and reason maintenance which is increasingly used for inference systems in Artificial Intelligence. The full dissertation is expected to be sent to the printers by February 1985. A paper was presented at a refereed, limited-participation Workshop on Non-monotonic Reasoning, sponsored by the American Association for Artificial Intelligence.

4.3.2 Other work on reason maintenance

Tomas Lund, until June 1984 employed at IDA in the AILAB, has been working on a modification of the reason maintenance algorithm (the central technology of WATSON and similar systems) to handle the case of numerical belief values, such as the probabilities or confidence factors used in statistical decision theory. A simple such algorithm was designed and informally proven correct. Unfortunately Tomas' licentiate thesis on this work was not yet finished when he left AILAB to work for EPITEC.

4.4 Other AILAB Projects.

During the years 82/83 and 83/84, the AILAB did not manage to create a critical-size natural language activity. Carl-Wilhelm Welin and Ludmila Ohlsson left the group for Ericsson. Still we think it is important to develop competence in this area, especially with respect to the use of the Swedish language e.g. in connection with expert systems. Mats Wiren is carrying on work in this area. He has produced a simple ATN system [Wiren83], and also supervised a student project which implemented a system for full analysis of Swedish morphology. Now we believe it possible to revitalize the natural language activity in AILAB.

4.5 Plans

Current projects include, in addition to those described more extensively above, expanded work on natural language interfaces (Wirén, Dahlbäck) and development of a constraint-based programming environment, ICONStraint, (Maleki). The practical end product of our coming activities will be some kind of knowledge representation system writing system (KRSWS). However AI rely heavily on practical experiments. In the not too far future we will consequently, again attempt an experiment in the "OBS" tradition.

The aim of the group is to build both software and theory. Observe that software construction is the experimental basis for the theory formation process in the AI-community. Observe that parts of AI theory does not need to be, and in some cases not even can be, formulated using mathematical formalisms. The most important strategy for the research will be experimental design.

The KRSWS activity is intended to function as a kernel project, around which AILAB's activities can be focused. The kernel project will serve two purposes. It will make it simpler to transfer AILAB's expertise out into industry. It will give new PhD candidates a basic knowledge of the ideas and techniques used within the laboratory.

4.6 AILAB working paper series.

Beside of the external publications, departmental reports and dissertations, AILAB had a working paper series. During the 1983 and the beginning of 1984, the following papers where issued.

83-01-11 **Grunditz, Hein and Lund:** KLAMP PART I.

83-01-10 **Maleki:** ORBIT programming manual

83-03-20 **Hein:** PAUL - the kernel of a representation and reasoning system for knowledge engineering tasks.

83-08-10 **Goodwin and Lund:** LYDIG project

83-12-06 **Wiren:** An ATN with small English and Swedish grammars.

In the future the use of working papers will probably be toned down, the idea being to write for report quality directly. The change is signalled by [Tengvald84a], [Tengvald84b] and [Goodwin84].

4.7 References:

[Goodwin84] WATSON: a dependency directed inference system. Report: LitH-IDA-R-84-08, Department of Computer and information Science, Linköping University, S-581 83 Linköping (To be presented at the AAAI workshop on non-monotonic reasoning) (1984)

[Grunditz83] KLAMP (part one), H. Grunditz, U. Hein, T. Lund, AILAB working paper no 12. (1983)

[Rich83] Artificial Intelligence, E. Rich, McGraw-Hill (1983)

[Snow64] The two cultures, C.P. Snow, 2:nd ed., London Cambridge U. (1964)

[Tengvald84a] Reducing design complexity, or why does AI work, E. Tengvald, forthcoming report: LitH-IDA-R-84-xx, Department of Computer and information Science, Linköping University, S-581 83 Linköping (Presented at the AIMSA-84 conference in Varna, Bulgaria) (1984)

[Tengvald84b] AI an emerging science, E. Tengvald, forthcoming report: LitH-IDA-R-84-xx, Department of Computer and information Science, Linköping University, S-581 83 Linköping. (1984)

[Wiren83] An ATN with small English and Swedish Grammars. AILAB working paper no 18. (1983)

5.

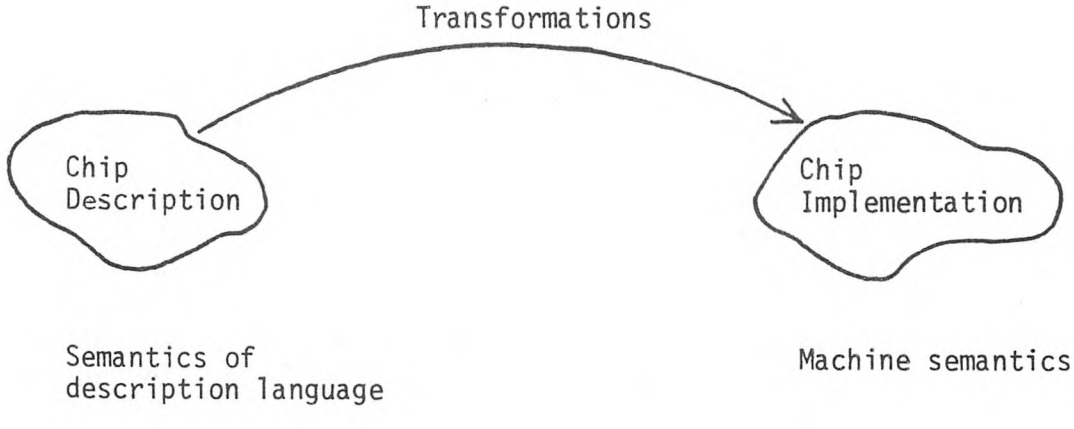
CADLAB
The Laboratory for Computer-Aided Design of Digital Systems

5.1 Introduction.

The laboratory for Computer Aided Design of Digital System, CADLAB, is concerned with the specification and construction of digital systems, especially those involving very large scale integrated circuits (VLSI).

CADLAB was formed from Professor Harold Lawson's Telesystem group when Telesystem, Datalogi and ADB merged to form IDA. In addition to its being part of IDA, CADLAB also cooperates with Professor Christer Svensson's group in IFM to form the VLSI Design Center.

The first years of VLSI Design Center were devoted to building competence and acquiring basic software. The year 83/84 marked the transition from the building phase to initiating new research. CADLAB is broadly concerned with the problem of silicon compilation; the process of translating a high level description of a system to a silicon layout. One model of the silicon compiler is that of a translator which takes a high level description of a chip and transforms the semantic content of the description into a machine:



We do not expect a single programming language to be equally useful for all applications, or a single machine architecture to run all applications equally well, or a single set of compiler optimizations to match all languages and machine architectures. We still have need for LISP, COBOL and ADA; for VAXs and CRAYs and IBMs. We should not expect a single hardware description language and its associated silicon compiler to solve all problems.

5.2 Current Work

CADLAB is currently engaged in a research project called ASAP (An Architectural Strategy for Asynchronous Processing) which is an attempt to provide an architectural basis for sophisticated CAD tools. Specificly we are interested in exploring the implications of asynchronous design and distributed control. Rather than attempting to compile systems that are general purpose, we are interested in special purpose systems such as are embedded in a wide variety of products such as telecommunication systems, electronic and biological instruments, robots and production control systems. The basic assumption is that in embedded, or special purpose systems, the structure of the application is well defined. In such environments we are faced with only a small number of programs and the payoff in being able to specialize the system is quite high.

5.3 VLSI layout and Timing Problems.

Shrinking VLSI geometries promise both denser chips and decreased switching times at the gate level. Unfortunately, while gate delays scale linearly, the RC-line delay for communications between gates does not scale, thus leading to a situation where the chip speed is limited by the interconnections. Even with today's technology "it takes about as long for a signal to cross a chip of side .5mm as it does to go along a coaxial cable 75cm long."

The constant propagation delays and decreasing gate delays mean that a system-wide clock whose rate is proportional to the gate delay is not possible if we are to maintain propagation delays at a small (five to ten percent) fraction of the clock period. However, within regions of a VLSI circuit, called isochronous or equipotential, the system may be considered synchronous at the maximum clock rate permitted by the circuit technology. The problem is then to synchronize independent isochronous regions. This may be implemented as either a self-timed discipline or based on a hierarchy of clocks.

The concept of isochronous regions on a chip is generalizable to that of a packaging hierarchy for computers. Conventional packaging groups circuits into chips, printed circuit boards, and cabinets. Current high performance technologies group the chips into modules and the modules into frames. Future technologies, including wafer scale integration, should be thought of as extensions of the goal of optimizing machine performance by minimizing communications costs.

What is lacking is a framework for a design system in which machines can be designed without regard to the implementation or packaging of their component modules and then either allow the designer to explore the performance and cost implications of different implementations or have the system attempt to automaticly provide an optimized solution from a library of function implementations.

5.4 Timing and the Exploitation of Natural Parallelism.

Typical tasks to be performed in embedded systems are data capture, processing, control signal generation, display maintenance and possibly statistics gathering. These heterogeneous tasks when implemented for a *processor* (which we shall abbreviate as *pr*) are viewed as *processes* (which we shall abbreviate to *ps*). For conventional general purpose *pr's* (including microprocessors), the *ps's* are programmed in a "language" suitable for the *pr* and mapped into the sequential execution domain of the *pr*. The treatment of timing idiosyncrasies of the application domain are placed under the jurisdiction of a real time operating system (also operating in the sequential execution domain of the *pr*). Thus, the potentially highly parallel environment of the application tasks is placed into the procrustean limits of the sequential execution of a single *pr* under an operating system with a potentially complicated interrupt and process management structure. Any potential parallelism implicit in the application tasks is thereby lost.

In embedded systems, it would be desirable to exploit the inherent and natural *parallelism* of the application tasks. In order to accomplish this property and to eliminate the complexities and eventual problems with centralized real time operating systems, we require asynchronous processing (at potentially very high levels) and distributed processing logic.

5.5 Extensibility and Specialization.

Instead of binding the solution of *ps* realization to a single general purpose *pr*, we establish the possibility of selecting the best specialized form of *pr* for accomplishing the task of the *ps*. A family of potential *pr* realization methods should be available in the library of the accompanying CAD system. In the simplest case, the *pr* could be a combinatorial network of logic and/or an analogue circuit. We shall term such logic as *base logic pr's*. Above this level, we introduce *programmed logic pr's* which in their simplest form could be programmable logic arrays (PLA's) or some form of structure resembling various microprogrammed architectures. Further, the notion of programmed logic *pr's* is to be interpreted in its widest meaning to include logic for finite state machines, extended finite state machines, as well as instruction set processors for concrete realizations of programmed logic at any level of abstraction.

One can potentially think of a specialized *pr* for each and every *ps* of the

application. This specialization may even be the case in certain realizations, however, when the nature of the problem (including timing constraints) permits, several *ps's* may indeed be multiplexed onto a single programmed logic *pr*. The individual *ps's* are suitably decomposed into the execution representation required by the *pr*. In this case, local scheduling logic for allocating the use of the *pr* must be provided.

By distributing the processing logic and utilizing the asynchronous approach to timing, extensibility is easier to attain. New processing tasks of *ps's* can be added without making new demands upon the processing capability of a single *pr* upon which all *ps's* are multiplexed. The extensibility is provided in the framework of the asynchronous processing strategy, by the use of generalized interprocess communications which can be automatically "tailored" to the requirements of the *ps's* of the embedded system.

5.6 Hardware - Software Tradeoffs.

By providing an environment where the selection of *ps:pr* pairings is based upon the criterion of finding the best *pr* for a given *ps*, and where timing is handled by the asynchronous strategy, we have an ideal environment for evaluating hardware-software tradeoffs. The nature and complexity of the algorithm of the *ps* will be the dominant factor. Simple algorithms may result in base logic *pr* solutions; whereas, complicated *ps* algorithms may require one or more levels of programmed logic *pr's*.

CAD tools must provide various forms of synthesis and support tools for the potential variations of programmed logic *pr's*. These tools must include simulation and evaluation mechanisms so that insight and potential expert advice can be extracted for local *ps:pr* mappings as well as for the entire set of *ps's* with associated *pr's* of potential embedded system realizations.

ASAP is concerned with the design of *systems* rather than single chips. With increasing clock rates as a result of decreasing gate delays, packaging occupies an increasingly important role in determining the performance of a computer system. Given performance goals for a system and packaging constraints such as limited real estate on a chip or printed circuit board, we want to find an optimal system.

ASAP is based on decoupling the specification of functions from the implementation of those functions. Asynchronous processing has been quite widely used in order to resolve problems of interfacing computing elements which operate at different rates of speed or for which a common clock cannot be established. Asynchronous processing decouples the implementation of a process from its communications.

5.7 Design Methodology and the Man-Machine Interface.

Designs utilizing previously defined parts and designs undertaken by groups benefit from an asynchronous design style since the design efforts are decoupled. However, it is important that the different design teams agree on the external behavior of the various parts. CADLAB is cooperating with the Computer Systems Laboratory at Uppsala in applying the formal design techniques developed for description of communications protocols to the description of integrated circuits.

5.8 Planned Activities

In relationship to fundamental architectural issues we plan to explore existing and develop new languages for expressing concurrent processing and develop strategies and algorithms for the simulation, partitioning and optimization of systems consisting of communicating asynchronous processes.

We also expect to continue the work on interactive structure editing started by Mikael Patel and Arne Jönsson in their licentiat thesis and by Yoshikazu Yamamoto (*Keio University*) during his visit to Linköping 1981-1983. We plan to exlore graphical means of representing interprocess communication as well as continuing activities in algorithms and programmed logic representation. As a further step towards improving the man-machine interface, we shall explore methods to integrate knowledge based systems into CAD systems.

5.9 Personnel.

Professor Harold W. Lawson Jr., Ph.D.
J. Bryan Lyles, Ph.D.
Britt-Marie Ahlenback, secr.
Johan Fagerstrom, MSE
Bjorn Fjellborg, MSE
Tony Larsson, MSE
Mikael Patel, Tech.Lic.
Zebo Peng, BS

5.10 Licentiate Theses

Vojin Plavsic, Interleaved Processing of Non-Numerical Data Stored on a Cyclic Memory.

Arne Jönsson and Mikael Patel, An Interactive Flowcharting Technique for Communicating and Realizing Algorithms.

6.

LIBLAB

The Library and Information Science Research Laboratory

6.1 Introduction

LIBLAB is a joint project of Linköping University Library and the Department of Computer and Information Science. The project has been initiated with the twofold aims of 1) creating a basis for continuing library research, which until now has not been available in Sweden, and of 2) pursuing research in areas that are central to libraries as well as providing a mutually benficial exchange with computer science.

Funding of LIBLAB for an initial three years, starting January 1, 1983, was provided by the Swedish Delegation for Scientific and Technical Information, DFI, (a body corresponding to the National Commission on Libraries and Information Science, NCLIS, in the US), on the basis of an application and a research program presented in a report, LiU-LIBLAB-R-1982:3, (1).

In the research program, there were two major themes, and one minor, proposed for the research of LIBLAB. The central theme is document description and representation, and the other major theme is users and library (systems). The minor theme is networking, especially questions of central vs. local handling.

6.2 Libraries, Catalogs, and Representations

A library is probably perceived by most people as a place where there is a collection of books that can be borrowed, without charge. This conception has ofcourse a validity based on experiences, and catches some of the essentials, but there are also many other descriptions that can be made, with more specificity or generality. In (1) a number of such models were presented as a background to the discussions of the program of research for LIBLAB.

Cataloging is traditionally described as consisting of three tasks. The first is description of the item, the second is choice of headings (access points which determine under which heading the item can be found in the linear file that a

catalog ususally is), the third is choice of form for names.

6.3 Activities and Projects

The first year, 1983, was largely spent on 1) further delimiting the broad research areas in the research program, 2) reviews of the literatures on AI, especially knowledgebased systems, cataloging and rules for cataloging, and online public access catalogs, and on 3) building competence in utilizing the resources of the department.

In the rest of this presentation the activities of the second year, 1984, will be discussed and the plans for 1985 will be outlined.

The various reports and working papers produced until the end of 1984 are listed in separate section.

6.3.1 Overview

The central theme, bibliographic description and representation, has been further delimited so that catalogs, cataloging rules and the representations used in catalogs are the focus of the studies.

A first attack has been made on cataloging rules with expert systems as the "weapon". (EMYCIN is available at the department and there is experience in applying it).

3RIPCAT, a project concerned 1) with the design of a library catalog made available to users through their own terminals, using an information retrieval system, and 2) the effects on the use of the catalog and the library, has been prepared.

A survey of the various forms of bibliographic representations, their elements and structure, and the transformations of these has been initiated.

Abstract editor formalism has been perceived as potentially relevant and will be used to build and test a formalized model for the bibliographic description of documents.

A vision of an alternative to the traditional catalog, called the HYPERCATalog, and based on associations and links, still in a tentative form, has been elaborated and will be the major project.

6.3.2 An Expert System for Simple Choices of Access Points from Entries.

A demonstration system, (which was gradually built in a number of versions, using EMYCIN), now called ESSCAPE (Expert System for Simple Choices of Access Points from Entries) has been developed, using a selection of the rules on the choice of access points in the Anglo-American Cataloging Rules, 2nd. Edition, (AACR 2).

Within LIBLAB there were a number of somewhat different goals with this approach. Firstly we wanted to get acquainted with expert systems and artificial intelligence. Secondly we hoped to get demonstrable "products" for the library community, one cataloging consultant for librarians and another for catalog users. Thirdly we hoped to be able to produce a simplified, reduced set of rules, giving cataloging records compatible with those from the complete rules, for all those who only occasionally do "cataloging". Fourthly it is interesting to look at an application of expert systems with rules as a domain. Lastly, but importantly, we think that expert systems provide a powerful instrument for revealing the structure and nature of some aspects of the cataloging rules.

There are three aspects of cataloging that have been started with. The first is what might be called building a relevant world. In it we concentrate on the first act of perception, the recognition of elements to be used in description, based on concepts and definitions involved in cataloging, (which is not covered by the rules), especially the difference between competence and performance. The second was straightforwardly to establish some subsets of the rules, e.g. headings, as a domain in an expert system and look at the problems involved in that, e.g. in what sequence to take the various rules, how to phrase questions, and justify decisions. The third is what the rules are trying to accomplish, i.e. to try to find out whether the same "effect" can be reached with differently organized and formulated rules.

The next step will be to backtrack, with the understanding gained of cataloging, and to attempt to build a first theory of bibliographic description and of representations for catalogs of documents.

6.3.3 3RIPCAT

In this project the goals were to establish an experimental online public access catalog using an IR-system (3RIP) (2) as its base. (This project has, however, not proceeded beyond the planning stage because LIBLAB's reference group considered the HYPERCATalog project to be more interesting, and that it hence should be given most of the resources.) The assumptions that we wanted to test were 1) that an IR-system, with the catalog file regarded as an example of a bibliographic database, is better as a basis for a public access catalog, than a database updating system, and 2) that access for everyone to the catalog through their own terminals, in their offices, will change the use of the catalog, the literature and the library.That would have repercussions for the design of the record and, eventually, for the description of documents, and thus for the cataloging rules.

The catalog was to be made available to all users having access to terminals, and in addition to studying the consequences for record structure etc., we were also interested in studying the effects on users and their uses of the library as well as the impact on the library, and the new demands encountered.

Parts of the 3RIPCAT-project will be continued in different forms. A first study of user behaviour in a library will be made by Hans-Ove Frid, who has a research scholarship from DFI. The file of reports acquired by the department will be transferred to the 3RIP-system and to a LISP-machine, to be used as a

test database in experiments.

6.3.4 Bibliographic representations and their transformations

Most of the people employed in a library do not handle the physical documents but rather various representations. A large part of the tasks performed in a library can be described in terms of handling of representations and transformations of representations.

A pilot study has been made of the representations encountered at the acquistions department and analyses are being done. Extended surveys will be made later.

6.3.5 Description of documents based on an abstract editor formalism

The rules for bibliographic description in the present cataloging rules are not based on any theory or formal model but have rather evolved to accomodate both the diverse types of documents and the limitations inherent in the traditional card catalog.

One of the problems in decription according to the present cataloging rules (e.g. AACR 2) is that no distinction is made between that which is based on "objectively" verifiable elements and that which is based on interpretation of these.

Based on the experiences gained in developing the ESSCAPE system and on a formalism for defining abstract editors (3) an attempt will be made to use such a formalism for describing documents, i.e. reconstructing them. It is envisaged that using this methodology a demarcation can be made between "pure" description and interpretation.

6.3.6 HYPERCATalog

During 1984 work was also initiated on a long term and large scale project called "HYPERCATalog", concerned with the structure and design of an enhanced catalog. HYPERCATalog is envisioned as being dynamic and containing much more information than present catalogs, especially in terms of links and relations between fields, records and files, and facilities for utilizing that. In addition to traditional searching, in which the user has to specify what he wants, the HYPERCATalog should support browsing and navigation as modes of using the catalog. Furthermore each user should be able to specify, and save, his views of the library. Three working papers (WP:9, WP:16, WP:18) describing the HYPERCATalog project in more detail are available.

In the first phase, which will be carried out during 1985, the features wanted will be defined, the problems that can be foreseen will be listed and a design specification will be elaborated.

The HYPERCATalog project is also planned to be a joint project, with participation from the Department of Library and Information Science at Tampere University in Finland, and Informatics Management and Engineering Ltd. in England. Additional funding for the joint parts has been applied for from NORDINFO.

6.4 Personnel

The singly most difficult problem for LIBLAB in initiating research in the library and informations science area in Sweden has been that of finding personnel, with a proper balance in the competences in library science and computer science. The more so since the interests and competences of the staff has an impact on the actual research performed, within the areas specified. LIBLAB is still not fully staffed, there is one vacancy for a doctoral student, but an interesting mixture has been achieved. The personnel of LIBLAB is briefly presented below, most of them participate in all projects but with different emphases.

Roland Hjerppe, MSE, Laboratory leader, will, apart from planning, coordination and administration etc., spend most of the time on the building of a model for bibliographic representation and on the HYPERCATalog project.

Bodil Gustafsson, BA, librarian, head of the cataloging department of Linköping University Library, works 50% of her time in LIBLAB, mostly on the project surveying the forms of representations, and on the HYPERCATalog project.

Hans Holmgren, MSE, divides his time equally between LIBLAB and teeaching at Administrative Data Processing, concentrating on the building of the model for bibliograhic description, and on the HYPERCATalog project.

Birgitta Olander, BA, librarian, former head of the acquisitions department, worked 50% of her time in LIBLAB on the ESSCAPE-project. BO is now at University of Toronto, Faculty of Library and Information Science, pursuing doctoral studies, and is expected to spend the later half of 1985 with LIBLAB in the HYPERCATalog project, before going back to finish her studies in Toronto.

Arja Vainio-Larsson, BA, psychologist, began her doctoral studies at LIBLAB the fall of 1984 and has mostly been taking courses but will participate mainly in building the model for bibliograhic description, and in the HYPERCATalog project.

Associated people:

The following have various associations to LIBLAB:

Kristian Wallin, student, responsible for local systems at the university library and for the NYTTFO-project in Linköping, will join LIBLAB as a doctoral student on a half-time basis after finalizing his BA-paper, and devote most of his time to the HYPERCATalog project.

Manny Jägerfeld, BA, who has a research scholarship from DFI for studying computerization in libraries, will also concentrate on HYPERCATalog.

Hans-Ove Frid, BA, who also has a research scholarship from DFI, but for studying catalog use, will be involved in the survey of representations and in the HYPERCATalog project.

6.5 List of publications

Reports:

(i.e. more extensive writings, reprints, etc.)

LiU-LIBLAB-R-1982:1
Hägglund, S. and Nordstrand, J-E. (Eds): Study on Directions for Research and Development of Scientific and Technical Information Systems. Febr. 1981, 53p.

LiU-LIBLAB-R-1982:2
Hägglund, S.: Informationshantering i det elektroniska kontoret. Juli 1982, 12p. (Also publisehed as LiTH-MAT-R-82-27, and in Proc. 5th Nordic Conf. Inf. and Doc., Trondheim, 1982)

LiU-LIBLAB-R-1982:3
Hjerppe, R,; Bivins Noerr, K. and Noerr, P.: A Program for LIBLAB, the library research laboratory at Linköping University. Sept. 1982, 49p.

LiU-LIBLAB-R-6
Hjerppe, R.: What artificial intelligence can, could, and can't, do for libraries and information services. Sept. 1983, 20p. (Also published in Proc. 7th IOLIM, Dec 6-8 1983, Learned Information Ltd. London.)

LiU-LIBLAB-R-7
Hjerppe, R.: LIBLAB 1983. Rapport till DFI över projekten "Laboratoriet för biblioteks- och informationsvetenskap. LIBLAB. FAS 1. Igångsättning." samt "FAS 2. Katalogregler, test av expertsystem." Dnr. 82-254 Maj 1984, 6p.

LiU-LIBLAB-R-9
Hedman, T.: Ämnessökning i bibliografiska databaser där dokumenten klassificerats enligt Klassifikationssystem för svenska bibliotek (SAB-systemet) Aug. 1984, 35p.

Working papers

(i.e. usually preliminary, smaller papers, distributed as requested and from separate mailing list)

1. *LiU-LIBLAB-WP:1* Hjerppe, R.: LIBLAB. Planer för verksamhetens inriktning. Ett diskussionsunderlag. Maj 1983, 10p.
2. *LiU-LIBLAB-WP:2* Hjerppe, R.: LIBLAB. Plan för verksamheten under resten av 1983. Sept. 1983, 3p.
3. *LiU-LIBLAB-WP:3* Hjerppe, R.: "Representation and Exchange of Knowledge as a Basis for Information Processes." IRFIS 5 - Fifth

International Research Forum in Information Science, 5- 7 september 1983, Heidelberg. Kort reseberättelse.

4. *LiU-LIBLAB-WP:4* (Transferred to LiU-LIBLAB-NOTES:1)
5. *LiU-LIBLAB-WP:5* Hjerppe, R.: Projektet Katalogregelanalys. Etapp 1 och 2. Okt. 1983, 6p.
6. *LiU-LIBLAB-WP:6* Hjerppe, R.: Några utgångspunkter för verksamheten vid LIBLAB. Nov. 1983, 11+1p.
7. *LiU-LIBLAB-WP:7* Hjerppe, R.: Telematik och LIBLAB Nov. 1983, 2p.
8. *LiU-LIBLAB-WP:8* Hjerppe, R.: Planer för verksamheten under 1984. Nov. 1983
9. *LiU-LIBLAB-WP:9* Hjerppe, R.; *Järvelin, K.:* Projekt HYPERCATalog Jan. 1984, 12+15p.
10. *LiU-LIBLAB-WP:10* Hjerppe, R.: SEMINARIESERIEN - Informationssökning och bibliotekskataloger. Jan. 1984, 2p.
11. *LiU-LIBLAB-WP:11* Olander, B.: Expertsystem för katalogiseringsregler - AACR2. Delrapport. Febr. 1984, 22p.
12. *LiU-LIBLAB-NOTES:5)*
13. *LiU-LIBLAB-WP:13* Hjerppe, R.; *Marklund, K.:* Biblioteksforskning, teoribildning och LIBLAB. (Published in Biblioteksbladet nr.5 1984) Mars 1984, 11p.
14. *LiU-LIBLAB-WP:14* Hjerppe, R.: Bibliotek, reaktiva eller interaktiva system? (Ett utkast för seminariet "Biblioteket i problemlösningsprocessen" i Vadstena 9 - 13 april 1984) April 1984, 6p.
15. *LiU-LIBLAB-WP:15* Hjerppe, R.: Biblioteken i informationssamhället (Några diskussionspunkter inför grupparbetena vid seminariet på Hanaholmen 25 - 28 april 1984) April 1984, 6p.
16. *LiU-LIBLAB-WP:16* Hjerppe, R.: HYPERCATalog projektet. Några förtydliganden. Maj 1984, 4+12p.
17. *LiU-LIBLAB-WP:17* Hjerppe, R.: Kort reserapport från "3rd Joint BCS and ACM Symposium on Research and Development in Information Retrieval" vid Kings College, Cambridge, 2-6 juli 1984 Juli 1984, 3+12p.
18. *LiU-LIBLAB-WP:18* Hjerppe, R.: HYPERCATalog projektet. Ytterligare förtydliganden och reviderad budget. September 1984, 5+1p.
19. *Liu-LIBLAB-WP:19* Hjerppe, R.: Behovet av lingvistiska hjälpmedel inom BDI-området. (En egen, (och några andras), modell(er) som underlag inför diskussion vid det av NAVFs EDB-senter for humanistisk forskning arrangerade mötet den 13-14 september 1984 i Stockholm) September 1984, 15p.
20. *LiU-LIBLAB-WP:20* Hjerppe, R.: Projekt 3RIPCAT. Utgångspunkter, frågeställningar och ansats. September 1984, 10p.
21. *LiU-LIBLAB-WP:21* Hjerppe, R.: Preliminary reflections on the

concepts of local - central in networking, and on resource sharing. Oktober 1984, 10p.

22. *LiU-LIBLAB-WP:22* Hjerppe, R.: LIBLAB. Plan för verksamheten 1985 Oktober 1984, 6+82p.

References:

[1] Hjerppe, R.; Bivins-Noerr, K. Noerr, P.: A program for LIBLAB, the library research laboratory at Linköping University. (Res. Report LiU-LIBLAB-R-1982:3, Sept. 1982)

[2] 3RIP - A System Manual. Version 4. (Paralog AB Stockholm 1983)

[3] Kimura, G.D.: Editing Abstract Document Objects.(Ph.D. Diss. Comp. Sc. Dept. Univ. Wash. Seattle, 1984)

7.

The Group for Theoretical Computer Science

The objectives of the group are the following:

(1) to contribute to the activities of the other groups by:
- preparing graduate courses presenting mathematical theories relevant for these activities;
- cooperating with selected projects.

(2) to continue own research on the topics listed below, in cooperation with foreign scientific centers.

7.1 Researchers and Projects.

The group has two subgroups, one oriented towards the theory of logic programs, especially their relationship to other formalisms such as attribute grammars or two-level grammars, and one group studying aspects of computational complexity. The intention is to have a supervisor and a few graduate students in each subgroup and otherwise to cooperate with projects in the other laboratories.

7.1.1 Group Members

	% 84/85	%83/84
Group leadership:		
Jan Maluszynski, PhD	65	60
Supervisors:		
Andrzej Lingas PhD	100	100
Employed graduate students:		
Christos Levcopoulos	85	--

7.2 Theory of Logic Programing

7.2.1 Relating Logic Programs and Attribute Grammars

(Jan Maluszynski).

This activity was initiated in October 1983 and is continued in cooperation with Pierre Deransart (INRIA - France). It is intended to apply in logic programming some methods developed for attribute grammars. For this purpose a relation between logic programs and attribute grammars has been examined. It was proved that any logic program can be transformed into a relational attribute grammar equivalent to this program in that sense that both define the same relation on the Herbrand universe of the program.

Preliminary results of the research are described in two reports. We got:

- a proof technique which may be used to prove some run-time properties of logic programs;
- a sufficient condition for a logic program under which an infinite term will be never created during its computation (i.e. a condition, under which the run-time occur-check can be avoided without any risk);
- a possibility of modelling data flow of a restricted class of logic programs by means of attribute grammars;
- a non-trivial class of logic programs which can be run with a simplified form of unification.

The intended continuation of this research should contribute to the methodology of compilation of logic programs.

7.2.2 Computational Complexity of Logic Programs

(Andrzej Lingas).

An elementary relation between the derivation length and the derivation depth of logic programs has been established. The relation gives an evidence that generally, the small depth of a logic program does not imply the existence of an efficient parallel implementation of the program. The results were presented at the Logic Programming Workshop in Portugal, 1983.

7.3 Computational Complexity

7.3.1 Geometric and combinatoric aspects of VLSI design

(Christos Levcopoulos, Andrzej Lingas).

Several problems arising in the VLSI chip design can be formulated in an abstract way as geometric and combinatoric problems. The research focuses on partitioning and covering planar figures with rectangles under the minimum

number or the minimum total edge length criterion. Also, related geometric problems of optimally partitioning planar figures into triangles or convex polygons have been intensively examined. Several efficient approximation heuristics for optimally covering or partitioning geometric figures have been designed. In a large part, the achieved results have been published in proceedings of international conferences, or submitted for publication. It is intended to pursue the research, and prepare a survey or a monograph on optimally covering geometric figures.

7.3.2 Graph Algorithms (Andrzej Lingas).

The research concentrates on designing efficient algorithms for computationally feasible instances of so called subgraph isomorphism problem. The problem consists in determining whether a graph is isomorphic to a subgraph of another graph, and has among others an application in organic chemistry to determine whether a chemical structure is a piece of another. In spite of the NP-completeness of several simple instances of this problem, an algorithm of moderate time complexity has been designed for subgraph isomorphism constrained to easily separable graphs of bounded valence.

7.4 Contacts within the Department

An important goal for the Theory Group is to add to the theoretical schooling of graduate students in the other laboratories.

7.4.1 Courses for Graduate Students.

The following courses have been held for graduate students:

Formal Language Theory (1983)

Algorithm Analysis and Complexity Theory (1983)

Semantics of Programming languages (1983)

Introduction to Logic (1984)

Mathematical Aspects of VLSI (1984)

Courses offered in academic year 1984/85

Attribute Grammars and Attribute Evaluators

Computational Complexity

Introduction to Logic Programming

The group has also assisted with lectures within the SOFT-2 and SOFT-3 courses (for participants from industry).

7.4.2 Direct contacts.

The major areas of interaction have been with PELAB (esp. regarding logic programming) and CADLAB (esp. regarding geometrical complexity; lately also parallel architectures).

7.5 Publications and conferences.

Since 1982 the members of the group published separately, or as co-authors 1 book, 1 Ph.D. thesis, 2 papers in international scientific journals (further 3 are submitted for publication), 12 papers in the proceedings of international conferences (one more is accepted for the 4th FST-TCS Conference in Bangalore) and 4 internal reports. The references to these publications can be found in the list of publications.

The group served as local organizing committee for the International Conference on Foundations of Computation Theory FCT-83 sponsored by EVA and the European Association for Theoretical Computer Science, in cooperation with the University of Bonn.

8.

ADP
Administrative Data Processing

8.1 Administrative data processing

Including management information systems analysis and information systems analysis and design.

The subject area covered by this group deals mainly with social aspects of design and use of software for administrative applications in private companies and public services. Essential problems are the transition from natural to formal languages and vice versa together with prerequisites for, constraints on, and effects of computerized support for activities where teamwork, personal judgement and experience traditionally have been, and are expected to be, of great importance. This topic comprises systems development and tools for analysis of information requirements and tools for prototyping, the drawing up of technical requirements specifications and other kinds of user-oriented documentation and evaluation of effects caused by the use of computerized systems. It does also contain - from a general point of view - social methodology for describing administrative professional activities, for implementation, maintenance and evaluation of user-oriented computerized support.

The undergraduate study programme for Systems Analysis takes the main part of the group's teaching efforts. Beyond that we give separate single-subject courses to the level of postgraduate studies as well as courses in other study programmes.

8.2 Research activities.

Post-graduate and research activities related to the ADP undergraduate programs cover mainly problems of formalization at the interface between formal logic and social science, including cognitive psychology. There is no formal subject-oriented research organization within the humanities and social sciences faculty (research is organized into interdisciplinary "themes"). This explains the comparatively small size of research activities within the ADP

group. Main research interfaces are towards other programmes, such as LIBLAB and ASLAB, especially office automation and statistical information systems.

8.3 Personnel:

Eva-Chris Svensson, MSc, director of undergraduate studies
Carina Björkman, secretary

Hans Holmgren, MScE
Kristo Ivanov, Ph D
Dan Strömberg, MScE
Per Övernäs, BSc

Appendix 1.

The Knowledge Transfer Program.

During the last years we have experienced a growing concern in industry about the rapid development in the information technology area and also a considerable increase in interest for what is going on at the university. For instance, the following observations are made:

* Software competence is becoming an increasingly critical resource
* Software costs pose serious problems
* Computerized systems are difficult to change and maintain
* There is a fast international development with joint programs for R&D

This development has resulted in a demand for a rapid expansion of educational programs, a pressure on university staff from the labor market, requests for direct assistance in industry projects and in general in an increased volume of contacts between industry and the university.

Thus we have made conscious efforts to improve our contacts with industry, especially to make knowledge transfer more effective. The outflow of personnel from our research program to industry position has been maintained at a reasonable level. We regret that some who leave, do it without finishing their degrees, but although the value of a licentiate or PhD is rapidly improving in industry, we still have to accept that the salary structure does not give the desired incentive to complete a degree in all cases.

One way for knowledge transfer is to accept commissions from industry in research-related projects. It is our opinion that we should be restrictive about undertaking such commissions in order to maintain the fundamental goals of a research department. Again the differences in salaries are such that we can not expect our personnel to stay at the university doing similar work as they might do in industry. Such commissions are then preferably forwarded to the consulting firms, which are rapidly establishing themselves in the university environment.

For our industry knowledge transfer program (KTP) we have chosen a third alternative. We have inaugurated a joint program where a small number of large industries which are heavily dependent on proficiency in information technology are invited to participate in knowledge transfer activities.

The goal for this program is to:

* Promote an effective use of the results from the STU program for knowledge development.

* Provide a knowledge base for industry.
* Secure the availability of qualified competence within novel information technology areas of high importance.
* Contribute to an awareness about industry needs within the university.

The fundamental assumption is that the university guarantee that research projects of a high international standard is carried out within areas of common interest. In connection with this research the university undertake to *organize projects for medium term visitors from industry with an emphasis on learning, technology evaluation and other forms of knowledge transfer.* The obligation for each participating company is to:

- assign one person full time or two persons half time working on the joint projects at the university, with the primary objective to learn and evaluate novel technologies and methods.
- Contribute 600 000 SEK a year to the KTP budget administered by the university.

The joint activities are organized in close contact with the research projects in the laboratories. Presently we have established two areas for the program:

- **Applications of expert systems,**
  including development support systems, methodology for knowledge acquisition and the relationship to more established techniques such as database management, interactive information systems and rapid prototyping etc. (ASLAB in cooperation with AILAB)
- **Production technology for software,**
  with an emphasis on programming environments, especially incremental tools for languages in the Algol/Pascal/Ada family. (PELAB)

We have approached companies both with an interest primarily in business applications and companies concerned with technical systems. The program started this summer and at present S-E-Banken (the largest Swedish bank), Ericsson Information Systems and ASEA are participating, each with two persons working halftime at the university. We expect to be able to accomodate a few more participants in the program within the near future and negotiations are presently carried out with interested companies.

The key ideas of the KTP effort are:

1. The university carries out research projects relevant for industry in aeras which are expected to have high future potential.
2. The program engages companies highly dependent upon advanced information processing.
3. The emphasis is on next-generation software technology.
4. Novel and advanced equipment and software tools are used in experimental settings.
5. The research content of the program should be of high international quality.
6. The ulitimate goal of joint activities is to supply participating companies with a qualified background for strategic decision making,

internal use, and internal training within the information technology area.

We feel that the following are the main benefits for the participating companies:

- The immediate availability of powerful enviroments for experimentation with new software technologies.
- Support for evaluation of new trends, methodologies and products.
- Sharing of resources, especially critical-size research teams in areas where competent personnel is a scarce commodity.
- Participation in pilot projects near the edge of the research front line.
- Education of own personnel.
- Basis for recruiting students after undergraduate education.

The program presents a highly efficient way of communicating results to industry and to provide immediate access to the international research community. We have also experienced that the demonstrated industry relevance of our research program improves the possibilities to recruit the best students for graudate education.

Appendix 2.

Graduate Study Program

Figure 1. below indicates the levels of degrees in the Institutes of Technology (i.e. schools of engineering) in the Swedish university system. The figures indicate the nominal numbers of years for the studies in each step.

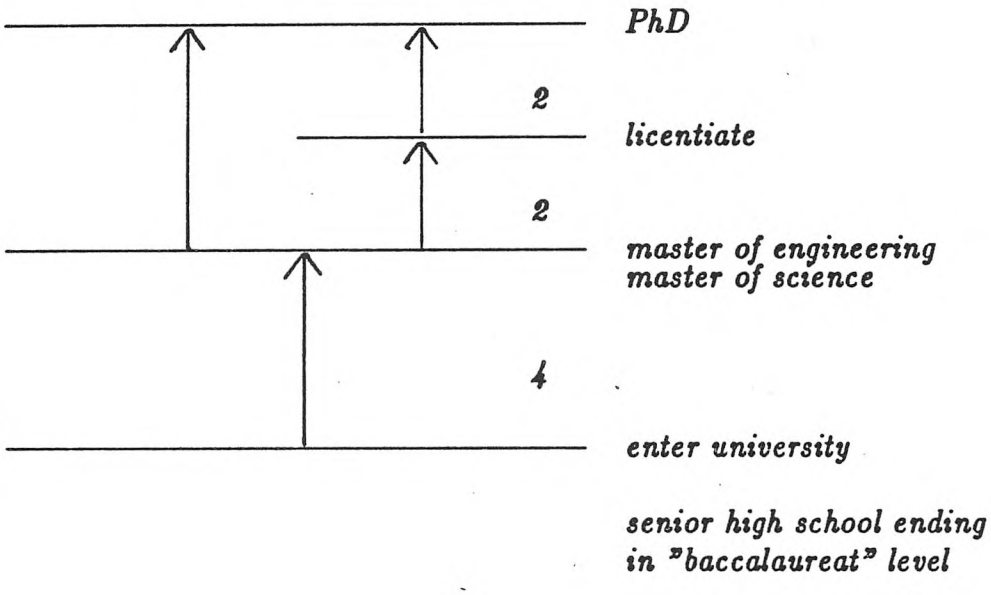


Fig 1. Levels of degrees

The graduate study program provides the studies from the level of master of engineering, *to* the licentiate and/or PhD degrees. The courses given by our department for the undergraduate education, up to the master's degree level, are described in appendix 3.

Graduate studies in the department of Computer and Information Science are organized as a program consisting of courses and project participation. The course program is planned at the department level and consists of *basic courses*, each of which is given every third year (if possible), and *occasional courses* which depend on the profile and interests of current faculty and visiting scientists. Project work is always done within one of laboratories or research groups.

Faculty engaged in graduate study program.

The list applies to the autumn semester, 1984

Pär Emanuelson, PhD (Linköping 1980, previous affiliation Uppsala), lecturer in computer science. Thesis supervision in PELAB. Functional languages, program verification, program analysis and program manipulation, programming environments, software engineering.

Peter Fritzson, PhD (Linköping 1984), researcher. Thesis supervision in PELAB. Tool generation, incremental tools, programming environments.

Anders Haraldsson, PhD (Linköping 1977, previous affiliation Uppsala), lecturer and director of undergrade studies in computer science. Thesis supervision in PELAB. Programming languages and systems, programming methodology, program manipulation.

Roland Hjerppe, (previous affiliation KTH, DFI and expert mission Tanzania) researcher. Group leader, LIBLAB. Library science and systems, citation analysis and bibliometrics, fact representation and information retrieval, hypertext, human-computer interaction and personal computing.

Sture Hägglund, PhD (Linköping 1980, previous affiliation Uppsala), researcher. Group leader, ASLAB. Database technology, human-computer interaction, artificial intelligence applications.

Kristo Ivanov, PhD (Stockholm 1973, since november,1984, appointed professor at Umeå), lecturer in administrative data processing. Systemeering methodology, psychology of problem solving, epistemological aspects of applications in social and economical/administrative sciences.

Harold W. Lawson Jr., PhD (Stockholm, several previous affiliations, also in industry), professor of telecommunication and computer systems. Computer architecture, VLSI, Computer-aided design, methodology of computer-related education and training.

Bengt Lennartsson, PhD (Göteborg 1974, previous affiliation Luleå), researcher. Group leader, PELAB. Programming environments, real-time applications, distributed systems.

Andrzej Lingas, PhD (Linköping 1983, previous affiliation Warszawa and MIT), researcher. Group leader in geometric complexity. Complexity theory, analysis of algorithms, geometric complexity, graph algorithms, logic programming, VLSI theory.

Bryan Lyles, PhD (Rochester 1982), acting professor (1984) of telecommunications and computer systems. Group leader, CADLAB. Computer architecture, VLSI, user interfaces, distributed systems.

Jan Maluszynski, PhD (Warszawa 1973, several previous affiliations), docent and group leader in theoretical computer science. Logic programming, software specification methods.

Erik Sandewall, PhD (Uppsala 1969), professor of computer science. Group leader in office systems. Representation of knowledge with logic, theory of information management systems, office information systems, autonomous expert systems.

Bo Sundgren, PhD (Stockholm 1973, previous affiliation Uppsala, also at Statistics, Stockholm), adj. professor. Group leader in statistical information systems. Database design and database-oriented systemeering, conceptual modelling, statistical information systems.

Erik Tengvald, PhD (Linköping 1984), researcher. Group leader, AILAB. Artificial intelligence, knowledge representation, planning and problem solving, expert systems.

Associated faculty in other departments:

Jan-Olof Brüer, PhD, researcher in information theory. Office information systems, especially security issues.

Ingemar Ingemarsson, PhD, professor of information theory. Information theory, security and data encryption, error correction codes and data compression.

Ove Wigertz, PhD, professor of medical informatics. Medical information systems, expert systems.

Graduate Study Course Program 1983-84

Basic and Occasional Graduate Courses:

Principles of Database Systems. (Sture Hägglund)

Conceptual Models and Database-Oriented Systemeering. (Bo Sundgren)

Mathematical Aspects of VLSI (Andrzej Lingas)

Introduction to Logic (Jan Maluszynski)

Formal Design Techniques for Communication Systems
(Björn Persson, Rune Gustafsson.)

Social Science Aspects of Information Processing (Kristo Ivanov)

Research-Related Courses and Seminars:

Office Models (Jan-Olof Bruer)

Non-Monotonous Logic (Jim Goodwin)

Object-oriented Representation Systems (Uwe Hein)

Theory for Information Management Systems (Erik Sandewall)

Graphical User Interfaces in Incremental Systems. (Bengt Lennartsson)

Testing Methodology in Incremental Systems. (Pär Emanuelson)

Information Retrieval and Library Systems. (Roland Hjerppe)

Interlisp-D Programming Methodology
(Anders Haraldsson, Sture Hägglund)

Graduate Study Course Program 1984-85

Basic and Occasional Graduate Courses:

AI and Expert Systems (Erik Sandewall)

Formal Methods in the Design and Verification of Microprogrammed Hardware. (Piotr Dembinski)

Distributed Computing. (Bryan Lyles)

Parallel Machines and Message-Based Architectures (Bryan Lyles)

Introduction to Logic Programming (Jan Maluszynski)

Sequential Algorithm Analysis and Computational Complexity (Andrzej Lingas)

Attribute Grammars and Attribute Evaluators (Pierre Derensart)

Information Retrieval and Artificial Intelligence (Linda C. Smith)

Dependency-Directed Reasoning (Jim Goodwin)

Computer Related Education and Training: Content and Methodology (Harold W. Lawson)

Edinburgh LCF (Jacek Leszczylowski)

Incremental Programming Systems (Bengt Lennartsson)

Research-Related Courses and Seminars:

Program databases (Bengt Lennartsson)

AI Programming. (Erik Tengvald)

Information Retrieval and Library Systems. (Roland Hjerppe)

Expert System Tools - comparative analysis and evaluation (Sture Hägglund)

Robotics of today and the future (Peter S. Nilsson)

Special Courses for the Knowledge Transfer Program:

The Structure and Interpretation of Computer Programs (Roland Rehmnert) (Fall and spring)

General Seminars and Visitors 1983-84

General seminars fall 1983

24/8 Don Walker: Natural Language and Knowledge Resource Management

30/8 Patricia Mainwaring-Samwell: Monitoring Concurrent Process Interactions

7/9 Computer Graphics Interfaces (Bengt Lennartsson). Presentations by Hans Marmolin, Foa, Else Nordhagen, Oslo, Ingvar Törnblom, Lars Kjelldahl and Yngve Sundblad, Stockholm.

20/9 Björn Pehrsson: CADDIE - a System for Computer-Aided Design

6/10 Mats Löfström: Design av ett textdatabassystem.

18/10 Bryan Lyles: HELM - Uniform Language Access to OS in a Network

10/11 Bo Ragnar Ohlsson: Metodik för utformning av människa-maskin dialog

22/11 Östen Oskarsson: Några funderingar kring projektstyrningsbegreppet.

29/11 Per Svensson: Cantor, implelementering av ett databasystem.

7/12 Konceptuell modellering vs. kunskapsrepresentation, inledare: Erik Malmborg

13/12 Mats S. Andersson: Datanät

14/12 Bo Sundgren: Statistiska informationssystem

General seminars spring 1984

24/1 Tomas Lund: Presentation av InterlispD och Xerox 1108

7/2 Leif Samuelsson, Mitzi Maynor: ISDOS, PSL/PSA and SEM. An environment for system development.

27/2 Piotr Dembinski: Software Tools in the Design and Verification of Microprogrammed Structures.

13/3 Peter Fritzson: Arkitekturen hos en inkrementell programmeringsomgivning och några konsistensbegrepp.

20/3 Klaus Kohnert: A Comparison Between KRL and MSRL

27/3 Peter Fritzson: Adaptiv prettyprintning av abstrakt syntax tillämpad på Ada och PASCAL.

3/4 Claes Höijenberg: Demonstration av Xerox 8000 STAR

12/4 J Sinddall: AI Applications in Computer-Aided Design

12/4 Warren Teitelman: Report on the Cedar Programming Environment Project

17/4 Dieter Hammer: Design of a Multiprocessor and its Associated Programming System

17/5 Alan Mycroft: Prolog Programs and Many-valued Logic

22/5 Istvan Szikszai: Applications of multi/micro-processors in data processing

5/6 Seif Haridi: Aspects of the OR-Parallel Token Machine: Storage Management and Control of Activities

20/6 David Douglas: Representation and Manipulation of Historical Models

21/6 Andrzej Proskurowski: Recognition of Partial k-trees

28/6 Pierre Deransart: Constructing Attribute Evaluators

27/7 Ken Pier: Design of the Xerox Dorado System

Foreign visitors

During 1983/84 we have received many visitors from industry och collegues from other universities for presentation and discussion of research and education. The following have been the foreign visitors:

Don Walker, SRI International, Palo Alto
Patricia Mainwaring-Samwell, London
Mokoto Nagao, Kyoto University,
Adolfo Guzman, UNAM University, Mexico City.
S. Uchida, S Shibayama, ICOT, Tokyo
Motoei Azuma, NEC, Yoshihiko Futamura, Hitachi,
and Tatsuro Matzuyama, Fujitsu, Tokyo.
Ann Schabas, University of Toronto.
Klaus Kohnert, Berlin
Jan Doroszewski, Warszawa
Hubert Dreyfus, Berkeley
Gerald Sussman, MIT
Istvan Szikzai, Budapest
Warren Teitelman, SUN, Palo Alto
Ken Pier, Xerox, Palo Alto
Andrzej Proskurowski, University of Oregon
Pierre Deransart, INRIA
Dieter Hammer, Academy of Science of the DDR
David Douglas, Edinburgh
Gerald M. Weinberg, Lamar Ledbetter, Lincoln Nebraska
Jan Komorowski, Harvard

Jean-Claude Latombe, Grenoble
Jörgen Born Rasmussen, San Diego
F.-L. Krause, Berlin
Tatiana Pateva, Ministry of Transport, Sofia
Donald Fairhall, Ballarat Coll., Australia
Belver Griffith, Drexel Univ., Philadelphia

Appendix 3.

Undergraduate Education

Undergraduate teaching in the school of engineering

The group for undergraduate teaching (the UDD-group) is responsible for courses in the two subjects *Computer Science* and *Telecommunication and Computer Systems* given in the undergraduate study programs at the Institute of Technology at Linköping. These curriculums are (figures give number of students accepted annually):

Computer Science (C) for 30 students
Computer Science and Technology (D) for 90 students
Industrial and Management Engineering (I) for 188 students
Mechanical Engineering (M) for 120 students
Applied Physics and Electrical Engineering (Y) for 180 students

These curriculums are four-year programmes and lead to a Master of Engineering or (for the C-programme) a Master of Science degree.

There are also single-subject courses from these programmes given as part-time and evening courses, and external courses given directly to companies and organizations. There is an ongoing discussion between a coalition of Swedish engineering industry (Oktogonen) and the Institute of Technology at Linköping, especially the department of computer science, to develop a programme for "continuing education" of engineers from these companies in computer science.

Courses. During 84/85 we will give a total of approx 50 different courses. In these curriculums we give 38 courses with a total of 3300 participants, 6 single-subject courses with 200 participants and about 4 - 6 external courses.

These curriculums have at least one introductory course in computer science and programming.

In the C- and D-programmes and in the variants towards computer science in the M- and Y-programmes (which students can choose after the second year) there are courses in

- **programming methodology**
- **assembly programming**
- **data structures**
- **data bases**
- **compiling techniques**
- **principles of programming languages**
- **concurrent programming**
- **operating systems**
- **computer networks**
- **computer architecture**
- **computer aided design of electronics**
- **discrete simulation**

The C-and D-programmes include two software projects. One individually the first year and one in a group during the third year. The projects require of both oral presentations and written reports.

In the C-programme we give in collaboration with other subjects a number of human-oriented courses:

- **linguistics I and II**
- **psychology, introductory course**
- **psychology of communication**
- **interactive systems**

We give also courses in theoretical computer science,
- **logic, introductory course**
- **formal languages and automata theory**
- **programming theory**
- **logic programming**

and courses in **artificial intelligence** (also given in the D-programme).

Computer facilities. A variety of computer systems are available to our students. Most courses use a DEC-20 computer running the TOPS-20 operating system at the Computer Centre at Linköpings University. It supports ca 60 terminals running concurrently.

The department has a PDP11/70 with UNIX for teaching purposes with 15 terminals, and some small computers (ABC80, ABC800 etc). A number of Apple's MacIntosh and LISP machines are also available for students in some courses.

There are 9 terminal rooms (8-9 terminals per room) and a network for connecting terminals to the various computer systems available for educational purposes.

Staff. The teaching is made by full or half time employed lecturers, by other persons with research appointment, by graduate students with teaching assistantships, and by the students themselves as part-time course assistents.

During 84/85 the staff consists of

5 full time and 3 half time senior lecturers (associate professors)
1 full time and 3 half time lecturers (assistant professors)
12 postgraduate students with 25% - 50% teaching assistantships
ca 15 other persons, professors and research assistents
ca 10 teachers from other subjects
ca 40-50 part-time course assistents

Personnel.

Anders Haraldsson, PhD, associate professor in computer science,
director of undergraduate studies
Barbara Ekman, secretary

The following persons are teaching one or more courses:

Computer Science:
Nils Dahlbäck, PhD
Pär Emanuelsson, PhD
Peter Fritzson, PhD
Christian Gustafsson,BSc
Anders Haraldsson, PhD
Sture Hägglund, PhD
Erland Jungert, PhD
Arne Jönsson, MSc
Jan Maluszynski, PhD
Kerstin Olsson, MSc
Tommy Olsson, MSc
Roland Rehmnert, MSc
Erik Sandewall, PhD

Ola Strömfors, MSc
Katarina Sunnerud, MSc
Olle Willen, MSc
Mats Wiren, MSc

Telecommunication and computer systems:
Harold W Lawson, PhD
Bryan Lyles, PhD
Mikael Patel, MSc

Listing of Undergraduate Course Program 1984-85

Datateknisk översiktskurs (C1, D1)
Datateknisk grundkurs (Y1)
Databaser (C3, D3)
Databaser (Y3, Y4, I4, Md4)
Logik, grundkurs (C1)
Psykologi grundkurs (C2)
Operativsystem (D3tk, D4, Y4, I4)
Kompilatorer och interpretatorer (Y4, I4)
Administrativ databehandling (D4, Y4, I4)
Cobol (D4, Y4, I4)
Logikprogrammering (C3)
Programmeringsteori (C3)
Processprogrammering (D3pv, D4, Md4, Y4, I4)
Operativsystemteori (C3, D3pv, M4d)
Programmering och projektarbete i Pascal (C1, D1)
Lagringssstrukturer (C2, D2, Md3)
Assemblyprogrammering (C2, D2, Md3, I4)
Programutvecklingsmetodik (D2, Md3)
Programutvecklingsmetodik och programmeringsprojekt II (D3)
Programutvecklingsmetodik och programmeringsprojekt C (C3)
Data och programstrukturer (C2, D3)
Datorspråk (C3, D3pv, D3al, I4)
Artificiell intelligens C (C3)
Artificiell intelligens D (d4)
Datastrukturer och maskinnära programmering (Y3, Y4)
Programmering Y (Y2)
Programutveckling (I1)
Datastrukturer och programutvecklingsmetodik (I2)
Datorsystem och programmering (M1)
Programmering i inkrementellt systen (C1)
Interaktiva system (C1, D3pv, D3tk)
Lingvistik I (C1)
Formella språk och automatateori (C2)
Lingvistik II (C2, C3)
Kommunikationspsykologi (C3)

Diskret simuleringsteknik (D3, Y3)
Datornät (D4, Y4, I4, Md4)
Datorarkitektur (D4, Y4)
Datorstödd elektronikkonstruktion (D4, Y4, I4)

Programutveckling (Fristående enstaka kurs Linköping)
Kompilatorteknik (Fristående enstaka kurs Linköping)
Databaser (Fristående enstaka kurs Linköping)
Programmering i Pascal (Fristående enstaka kurs - distans)
Programmering i Ada (Fristående enstaka kurs Norrköping)
Programmering i Ada (Fristående enstaka kurs Linköping)

Undergraduate course program in systems analysis, in the school of humanities and sciences

The educational programme for systems analysis ranges over three years of fulltime studies. This aims at professional activities of design, evaluation and implementation of computer-based information systems. Because of that, ADP-systems analysis dominates the programme. Nevertheless great importance has been attached to other subjects in order to give the programme the necessary breadth and also to ensure that the students will become aware of the complexity of the community where computers can be used.

The first two years of the programme constitute a common core of basic studies for all students. Within the subject of ADP-systems analysis there are courses in systems development and systems theory as well as courses in programming and computer science. The courses about systems development and systems theory deal with formal methods and prototyping. For the programming courses Pascal has been chosen as the main language but, of course, other languages are taught as well. Within the field of computer science the students take courses in database design, development of interactive systems, methodology for program development, communication, evaluation of computer systems, programming methodology, etc. Other subjects that are given within the common core of basic studies are:

- business economics and management, to get basic knowledge about the organization of corporations and public services and their "commonday" routines.
- human factors, industrial and social psychology, including ergonomics, work environment, co-determination and participative management, group dynamics etc.

There are also courses in practical swedish language for professional use, social science, matematics and statistics. The second year ends with about five months of on the job training.

During the last year the students can choose between one of the following three specializations:

- Methods for data analysis (data analysis), aimed at statistical methodology and statistical analysis methods. This specialization includes documentation and presentation of projects where storage and retrieval of data are crucial.
- Development of computer programs and program systems (program development) aimed at program development, methodology and technology. This specialization contains courses about operating systems, compilers, interpreters etc.
- Development of information systems (systemeering), aimed at methodology for design and evaluation of information systems. The program includes in-depth studies of budgeting and accounting and their relation to project management and systems budgeting.

All three specialisations end with a term-paper reporting the development and implementation of an individual project.

Appendix 4.

Publications 1980-84

DISSERTATIONS 1980-84:

(*Linköping Studies in Science and Technology. Dissertations.*)

No 51 **Erland Jungert:** Synthesizing Database Structures from a User Oriented Data Model, 1980.

No 54 **Sture Hägglund:** Contributions to the Development of Methods and Tools for Interactive Design of Applications Software, 1980.

No 55 **Pär Emanuelson:** Performance Enhancement in a Well-Structured Pattern Matcher through Partial Evaluation, 1980.

No 58 **Bengt Johnsson, Bertil Andersson:** The Human-Computer Interface in Commercial Systems, 1981.

No 69 **H. Jan Komorowski:** A Specification of an Abstract Prolog Machine and its Application to Partial Evaluation, 1981.

No 71 **René Reboh:** Knowledge Engineering Techniques and Tools for Expert Systems, 1981.

No 77 **Östen Oskarsson:** Mechanisms of Modifiability in Large Software Systems, 1982.

No 94 **Hans Lunell:** Code Generator Writing Systems, 1983.

No 97 **Andrzej Lingas:** Advances in Minimum Weight Triangulation, 1983.

No 109 **Peter Fritzson:** Towards a Distributed Programming Environment based on Incremental Compilation, 1984.

No 111 **Erik Tengvald:** The Design of Expert Planning Systems. An Experimental Operations Planning System for Turning, 1984.

LICENTIATE THESES 1984:

(*Linköping Studies in Science and Technology. Theses.*)

No 17 **Vojin Plavsic:** Interleaved Processing of Non-Numerical Data Stored on a Cyclic Memory. 1983.

No 28 **Arne Jönsson, Mickael Patel:** An Interactive Technique for Communicating and Realizing Algorithms. 1984.

No 29 **Johnny Eckerland:** Retargeting of an Incremental Code Generator. 1984.

EXTERNAL PUBLICATIONS.

(*Papers published in books, journals or international conference proceedings.*)

1. **Johan Elfström, Jan Gillqvist, Hans Holmgren, Sture Hägglund, Olle Rosin, Ove Wigerts:** A Customized Programming Environment for Patient Management Simulations. *Proc. of the 3rd World Conf. on Medical Informatics*, Tokyo, 1980.

2. **Pär Emanuelson, Anders Haraldsson:** On Compiling Embedded Languages in Lisp. *Proc. of the 1980 LISP Conf.*, Stanford, Calif, 1980.

3. **Pär Emanuelson:** From Abstract Model to Efficient Compilation of Patterns. *Proc. of the 5th Int. Conf. on Programming*, Turin, 1982. Revised version to appear in *Science of Computer Programming.*

4. **Johan Fagerström, Bryan Lyles, Zebo Peng:** Naming Services in a Distributed Computer Architecture. Nordic Symposium on VLSI in Computers and Communications, June 13-15 1984, Tampere, Finland.

5. **Peter Fritzson:** A Systematic Approach to Advanced Debugging through Incremental Compilation. *Proc of the ACM SIGSOFT/SIGPLAN Symposium on High-Level Debugging, Pacific Grove, CA., March 1983.*

6. **Peter Fritzson:** Symbolic Debugging through Incremental Compilation in an Integrated Environment. *The Journal of Systems and Software* 3, 285-294, (1983).

7. **Peter Fritzson:** Preliminary Experience from the DICE System - A Distributed Incremental Compiling Environment. *Proc. of the ACM SIGSOFT/SIGPLAN Symposium on Practical Software Development Environments, Pittsburgh, PA. April 1984.*

8. **James W. Goodwin:** Why Programming Environments Need Dynamic Data Types. *IEEE Trans. Software Eng., vol SE-7,* no 5, 1981. Also in Barstow et al. (eds.) *Interactive Programming Environments*, McGraw-Hill, 1984.

9. **James W. Goodwin and Uwe Hein:** Artificial Intelligence and the Study of Language. *Journal of Pragmatics, 6*, pp 241-280, North-Holland, 1982.

10. **James W. Goodwin:** WATSON - A Dependency Directed Inference System, to appear in *Proc. of the AAAI Workshop on Non-Monotonic Reasoning*, New Palz, NY, 1984.

11. **Sture Hägglund:** Dialogue Models for Human-Computer Communication. A Practitioner's View. in *Proc. of the Workshop on Models of Dialogue: Theory and Application.* Linköping 1981.

12. **Sture Hägglund, Johan Elfström, Hans Holmgren, Olle Rosin, Ove Wigerts:** Specifying Control and Data in the Design of Educational Software. *Computers & Education, vol 6*, no 1, 1982.

13. **Sture Hägglund, and Roland Tibell:** Multi-Style Dialogues and Control Independence in Interactive Software. In Green et al. (eds.) *The Psychology of Computer Use*, Academic Press, 1983. Previous version in *Proc. of the 1st European Conf. on Cognitive Engineering*, Amsterdam, 1982.

14. **Sture Hägglund:** On the Design of a Query Environment for Office Use. *Proc. of the 2nd Scandinavian Seminar on Information Modelling and Database Management*, Tampere, 1983.

15. **Uwe Hein:** Interruptions in Dialogue. Also in D. Metzing (ed), *Dialogmuster und Dialogprozesse.* Hamburg, Buske, 1981.

16. **Uwe Hein:** Natural and Artificial Communications. - Some Reflections -. in *Proc. of the Workshop on Models of Dialogue: Theory and Application.* Linköping 1981.

17. **Uwe Hein:** Constraints and Event Sequences. *Proc of the NATO symp. on Artificial Intelligence*, Lawrence Erlbaum, 1982.

18. **Uwe Hein:** PAUL - A Programming Language for Knowledge Engineering Applications. *Proc. of the International Conference on Artificial Intelligence,*

Leningrad, October 1983.

19. **Roland Hjerppe:** What artificial intelligence can, could, and can't, do for libraries and information services. *Proc. 7th IOLIM,* Learned Information Ltd. London. December 1983.

20. **Hans Karlsson, Roland Lindvall, Olle Rosin, Erik Sandewall, Henrik Sörensen and Ove Wigerts:** Experience from Computer Supported Prototyping for Information Flow in Hospitals. *Proc. of the ACM SIGSOFT Second Software Engineering Symposium: Workshop on Rapid Prototyping,* Columbia, Maryland, April 19-21, 1982.

21. **Hans (Karlsson) Gill, Bertil Kågedahl, Erik Sandewall, Henrik Sörensen, Lennart Tegler, and Ove Wigerts:** A Notation for Information Flow Models Supporting Interactive System Development. *Proc of the 6th Annual Symposium on Computer Applications in Medical Care,* Washington DC, nov 1982.

22. **H. Jan Komorowski:** QLOG - The Software for Prolog and the Logic Programming. *Proceedings of the Logic Programming Workshop,* Debrecen, Hungary, 1980. Also in Clark, Tärnlund (eds.) *Logic Programming,* Academic Press, 1982.

23. **H. Jan Komorowski:** Partial evaluation as a means for inferencing data structures in an applicative language: a theory and implementation in the case of Prolog. *Proc of the Symp. on Principles of Programming Languages,* Albuquerque, 1982.

24. **H. Jan Komorowski:** An Abstract Prolog Machine. *Proc. of the European Conf. on Integrated Interactive Computing Systems,* Stresa, 1982.

25. **H. Jan Komorowski:** A Prototype Compiler for Prolog. Poster version presented at *the 6th Int. Conf. on Software Engineering,* Tokyo, 1982.

26. **Harold W. Lawson Jr.:** Computer architecture education, a chapter in Tiberghien (Ed.): *New Computer Architectures,* pp 224-285, Academic Press, 1984.

27. **Harold W. Lawson Jr.:** Impact of CAD and Integrated Circuit Developments on Telecommunication. *Proc. of the EUTECO Conference,* Oct 3-6 1983, Varese, Italy.

28. **Harold W. Lawson Jr.:** Ingrediants and Implications of Tomorrows CAD Systems. *Integrated Circuit Seminar,* July 18-22 1983, Singapore.

29. **Harold W. Lawson Jr.:** Architecting VLSI Systems. *Integrated Circuit Seminar,* July 18-22 1983, Singapore.

30. **Harold W. Lawson, Jr.:** Addressing Fundamental Problems in Computer Related Education and Training. To appear in *Proc. of the 4th World Conf. on Computers in Education,* Norfolk, 1985.

31. **Harold W. Lawson Jr., Bryan Lyles:** An Architecturial Strategy for Asynchronous Processing. *IFIP Workshop,* March 26-28 1984, Bristol.

32. **Christos Levcopoulos, Andrzej Lingas:** Covering Polygons with Minimum Number of Rectangles, *Proceedings of the STACS Symposium, Paris (1984), Lectures Notes in Computer Science, vol 166,* Springer Verlag.

33. **Christos Levcopoulos:** On Covering Regions with Minimum Number of Rectangles, *Proceedings of the Workshop on Parallel Computing and VLSI,* Amalfi, Italy, (1984) North-Holland Publ. Co.

34. **Christos Levcopoulos, Andrzej Lingas:** Bounds on the Length of Convex Partitions of Polygons, to appear in *Proceedings of the 4th FST-TCS Conference,* Bangalore, India, (1984).

35. **Andrzej Lingas:** Heuristics for Minimum Edge Length Rectangular Partitions of Rectangular Partitions of Rectilinear Figures, *Proceedings of 6th GI Conference on Theoretical Computer Science,* Dortmund (1983), *Lectures Notes in Computer Science,* Springer Verlag.

36. **Andrzej Lingas:** An Application of Maximum Bipartite C-Matching to Subtree Isomorphism, *Proceedings of the 8th Colloquium on Trees in Algebra and Programming,* L'Aquila (1983). *Lectures Notes in Computer Science, vol 159,* Springer Verlag.

37. **Andrzej Lingas:** A Note on Complexity of Logic Programs, *Proceedings of the Logic Programming Workshop*, Aldeia das Acoteias, Portugal (1983).

38. **Andrzej Lingas:** The Greedy and Delauney Triangulations are not bad in the average case and Minimum Weight Geometric Triangulation of Multi-Connected Polygons is NP-complete, *Proceedings of the International Conference on Foundations of Computation Theory*, Borgholm (1983), *Lecture Notes in Computer Science, vol 158*, Springer Verlag.

39. **J. Bryan Lyles:** CAD Approaches for an Asynchronous Architecture. Also in *Proc. of Nordic Symposium on VLSI in Computers and Communications*, June 13-15, 1984, Tampere, Finland.

40. **J. Bryan Lyles, Zebo Peng, Johan Fagerström:** Naming Services in a distributed Computer Architecture. Also in *Proc. of Nordic Symposium on VLSI in Computers and Communications*, June 13-15, 1984, Tampere, Finland.

41. **Jan Maluszynski, Jorgen Fischer Nilsson:** A Comparison of the Logic Programming Language Prolog with Two-Level Grammars. *Proc. of the 1st Logic Programming Conference*, Marseille-Luminy, 1982.

42. **Jan Maluszynski, Jorgen Fischer Nilsson:** A version of Prolog based on the notion of two-level grammar. *Proc. of the Prolog Programming Environments Workshop*, Linköping, 1982.

43. **Jan Maluszynski, Jorgen Fischer Nilsson:** Grammatical Unification. *Information Processing Letters, vol 15*, pp 150-158, (1982).

44. **Jan Maluszynski:** Towards a Programming Language based on the Notion of Two-Level Grammar. *Theoretical Computer Science, vol 28*, pp 13-43, North-Holland (1984).

45. **Ludmila Ohlsson:** A Computer Model for Domain Dependent Systems. *Proc of 7th Int. ALLC Symp. on Computers in Literary and Linguistic Research*, Pisa, 1982 (North-Holland).

46. **Gunter Riedewold, Jan Maluszynski, Piotr Dembinski:** *Formale Beschreibung von Programmiersprachen*, R. Oldenburg Verlag, Munchen, Wien, (1983).

47. **Olle Rosin, Hans Holmgren, Sture Hägglund**, Implementing Tuning and Feedback Facilities in a System for Patient Management Simulations, *Proc. 3rd Congress on Medical Informatics Europe*, Toulouse, 1981.

48. **Erik Sandewall et al:** Provisions for Flexibility in the Linköping Office Information System, *Proc. of the National Comp. Conf.*, Los Angeles, 1980.

49. **Erik Sandewall, Claes Strömberg, Henrik Sörensen:** Software Architecture Based on Communicating Residential Environments. *Proc. of the 5th Int. Conf. on Software Engineering*, San Diego, 1981. Also in Barstow et al. (eds.) *Interactive Programming Environments*, McGraw-Hill, 1984.

50. **Erik Sandewall, Henrik Sörensen, Claes Strömberg:** A System of Communicating Residential Environments. *Proc. of the 1980 LISP Conf.*, Stanford, Calif, 1980

51. **Erik Sandewall:** Unified Dialogue Management in the Carousel System. *Proc. of the ACM Conference on Principles of Programming Languages*, Albuquerque, NM, 1982. Appeared in print in *N. Naffah (ed.) Office Information Systems,* North Holland, 1982.

52. **Erik Sandewall:** An Environment for Development and Use of Executable Application Models. Presented at the *seminar "Software factory experiences"*, Capri, May 3-7, 1982.

53. **Erik Sandewall, Sture Hägglund, Christian Gustafsson, Lennart Jonesjö, Ola Strömfors:** Stepwise Structuring - A Style of Life for Flexible Software. *Proc. of the National Computer Conference,* Anaheim, 1983.

54. **Erik Sandewall:** Formal Specification and Implementation of Operations in Information Management Systems. In: Jan Heering and Paul Klint (eds.), *Colloquium Programmeeromgevingen*, MC Syllabus, Mathematisch Centrum, Amsterdam 1983.

55. **Piotr Siemienski:** A specialized VLSI CAD DATABASE. *Nordic Symposium on VLSI in Computers and Communications*, June 13-15 1984, Tampere, Finland.

56. **Dan Strömberg, Peter Fritzson:** Transfer of Programs from Development to Runtime Environments. *BIT, vol 20*, no 4, 1980.

57. **Ola Strömfors, Lennart Jonesjö:** The Implementation and Experiences of a Structure-Oriented Text Editor. *Proc of the ACM SIGPLAN/SIGOA Symposium on Text Manipulation*, Portland, Oregon, June 8-10, (SIGPLAN NOTICES, vol 16, no 6) 1981.

58. **Erik Tengvald**, Reducing Design Complexity, or Why does AI Work, *Proc. of the AIMSA-84 Conf.*, Varna, Bulgaria, (1984).

59. **Ove Wigerts, Johan Elfström, Sture Hägglund and Olle Rosin:** Computer-Assisted Training in Patient Management and Clinical Decision Making. in Pages et al. (eds.) *Meeting the Challenge: Informatics and Medical Education*, North-Holland, 1983.

60. **Jerker Wilander:** An interactive programming system for Pascal. *BIT, vol 20*, 2, 1980. Also in Barstow et al. (eds.) *Interactive Programming Environments*, McGraw-Hill, 1984.

OTHER RESEARCH REPORTS 1980-84:

(*Departmental reports, contributions to nordic conferences, and papers awaiting external publication.*)

61. **Pierre Deransart, Jan Maluszynski:** Modelling Data Dependencies in Logic Programs by Attribute Schemata, INRIA Report RR323, (1984).

62. **Pierre Deransart, Jan Maluszynski:** Relating Logic Programs and Attribute Grammars. Submitted for publication (1984).

63. **Pierre Deransart, Jan Maluszynski:** Modelling Data Dependencies in Logic Programs by Attribute Schemata. LiTH-IDA-R-84-08

64. **Kenth Ericson, Hans Lunell:** Redskap för kompilatorframställning LiTH-MAT-R-80-39

65. **Peter Fritzson:** Distribuerad PATHCAL: Förslag till ett distribuerat interaktivt programmeringssystem för PASCAL. LiTH-MAT-R-81-05

66. **Peter Fritzson:** Fine-Grained Incremental Compilation for Pascal-Like Languages. LiTH-MAT-R-82-15

67. **Peter Fritzson:** Adaptive Prettyprinting of Abstract Syntax applied to ADA and PASCAL. LiTH-IDA-R-83-08

68. **Peter Fritzson:** The Architecture of an Incremental Programming Environment and some Notions of Consistency. LiTH-IDA-R-84-02

69. **James W. Goodwin:** An Improved Algorithm for Non-monotonic Dependency Net Update. LiTH-MAT-R-82-23

70. **Hans Grundits, Uwe Hein, Erik Tengvald:** Artificiell intelligens i framtidens CAD/CAM system. LiTH-MAT-R-82-30

71. **Sture Hägglund:** Towards Control Abstractions for Interactive Software. A Case Study. LiTH-MAT-R-80-37

72. **Sture Hägglund** et al: 80-talets elektroniska kontor: Erfarenheter från LOIS-projektet. LiTH-MAT-R-81-04

73. **Sture Hägglund:** Informationshantering i det elektroniska kontoret. *Proc. of the 5th Nordic conf. on Information and Documentation*, Trondheim, 1982. LiTH-MAT-R-82-27.

74. **Sture Hägglund:** En analys av interaktiva frågesystem för relationsdatabaser. *Proc. NordData 83, Oslo, 1983.*

75. **Anders Haraldsson:** Experiences from a Program Manipulation System. LiTH-MAT-R-80-24

76. **Anders Haraldsson:** INTERLISP - en avancerad integrerad programmeringsomgivning för LISP-språket. *Proc. Nord-Data 82*, Göteborg, 1982. LiTH-MAT-R-82-29.

77. **Kristo Ivanov:** From Computers to Information and Systems Science. LiU-MAT-ADB-R-80-3

78. **Kristo Ivanov:** Teologisk logik och systemteori. LiU-MAT-R-81-2

79. **Kristo Ivanov:** Sekundära sannolikheter i beslutsfattande. LiU-MAT-R-81-3

80. **Kristo Ivanov:** An elementary data-structure for data processing systems. LiU-MAT-R-82-2

81. **Kristo Ivanov:** Presuppositions of formal methods for development of computer systems. LiU-IDA-R-83-1

82. **Kristo Ivanov:** Computer applications and organizational disease. LiU-IDA-R-83-2

83. **Kristo Ivanov:** Systemutveckling och ADB-ämnets utveckling. LiU-IDA-R-84-1

84. **Erland Jungert:** Deriving a Database Schema from an Application Model Based on User-defined Forms. LiTH-MAT-R-80-35

85. **H. Jan Komorowski, James W. Goodwin:** Embedding Prolog in Lisp: An Example of a Lisp Craft Technique. LiTH-MAT-R-81-02

86. **Hans Lunell:** Some notes on the terminology for Compiler-Writing Tools LiTH-MAT-R-80-41

87. **Hans Lunell:** En konceptuell maskin för Pascal. (Preliminär version). LiTH-MAT-R-82-09

88. **Alexander Ollongren:** On the Implementation of Parts of Meta-IV in Lisp. LiTH-MAT-R-81-07

89. **Östen Oskarsson:** Construction of Customized Programming Languages. LiTH-MAT-R-81-10

90. **Östen Oskarsson, Henrik Sörensen:** Integrating Documentation and Program Code LiTH-MAT-R-81-01

91. **Ralph Rönnquist, Erik Sandewall:** The Relationship between Ordered and Unordered Trees in I.M.S. Theory. LiTH-IDA-R-84-04

92. **Kristian Sandahl:** An Experimental Evaluation of EMYCIN as a Tool for Implementation of Expert Systems. Report LiTH-IDA-EX-83-01.

93. **Erik Sandewall:** An Approach to Information Management Systems. LiTH-MAT-R-82-19

94. **Erik Sandewall:** Ny teknologi i kontorsdatasystem. *Proc. Nord-Data 82*, Göteborg 1982. LiTH-MAT-R-82-17.

95. **Erik Sandewall:** Partial Models, Attribute Propagation Systems and Non-Monotonic Semantics. LiTH-IDA-R-83-01

96. **Erik Sandewall:** Theory of Information Management Systems. LiTH-IDA-R-83-03

97. **Anders Ström:** DSS - ett datalagringssystem. *Proc. Nord-Data 82*, Göteborg 1982.

98. **Dan Strömberg:** Text editing and incremental parsing. LiTH-MAT-R-82-34

99. **Erik Tengvald:** En Intuitiv Förklaring till Kildalls Algoritm LiTH-MAT-R-80-27

100. **Erik Tengvald:** A Note Comparing Two Formalizations of Dataflow Algorithms. LiTH-MAT-R-80-28

101. **Erik Tengvald:** AI an Emerging Science. LiTH-IDA-R-84-11

102. **Lars Wikstrand, Sture Hägglund:** A System for Program Analysis and its Application as a Tool for Software Development and Program Transfer. LiTH-MAT-R-80-30

103. **Jerker Wilander:** Felkorrigering i inkrementella programmeringsomgivningar. *Proc. Nord-Data 82*, Göteborg 1982.

General:

104. **Anders Beckman:** Varför jag inte kan vara datalog: en diskussion av värderingar. LiTH-MAT-R-80-40

105. **J-O Brüer, S. Chowdhury, A. Fäldt, H. Gill and R.Rönnquist:** Office Models. ASLAB Memo 84-01.

106. **Andrzej Blikle:** Notes on the Mathematical Semantics of Programming Languages. (Lecture notes.) LiTH-MAT-R-81-19

107. **Pär Emanuelson:** Programtransformationer. LiTH-IDA-R-83-06

108. **Sture Hägglund, Jon-Erik Nordstrand** (eds.): A Study on Directions for Research and Development of Scientific and Technical Information Systems. (With contributions from **Hein, Hägglund** and **Sandewall.**)

109. **Sture Hägglund:** Datorstödda Informationssystem i ett regionalpolitiskt perspektiv. I Snickars (ed.) *Beslut för regional förnyelse*, Publica 1984.

110. **Sture Hägglund:** Kunskapsbaserade expertsystem. Ny teknik för applikationsutveckling i nästa generations programvarusystem. LiTH-IDA-R-83-07

111. **Uwe Hein:** A Proposal for an Artificial Intelligence Laboratory at SSRC, Linköping, 1980.

112. **Uwe Hein:** Vad är artificiell intelligens? LiTH-MAT-R-81-13 och tidskriften DATA, Köpenhamn, 1982.

113. **Uwe Hein:** Kunskapsteori: Representation, Manipulation och Organisation av kunskap - Del 1 - den teoretiska ramen. (Lecture notes). LiTH-MAT-R-82-04

114. **Uwe Hein:** Kunskapsteori: Representation, Manipulation och Organisation av kunskap - Del 2 - associativa nätverk. (Lecture notes). LiTH-MAT-R-82-07

115. **Uwe Hein:** Kunskapsteori: Representation, Manipulation och Organisation av Kunskap - Del 3 - Lingvistiskt orienterade representationssystem. (Lecture Notes). LiTH-MAT-R-83-08

116. **Uwe Hein, Sture Hägglund** (eds.): Proceedings of the Workshop on Models of Dialogue. Theory and Application, Linköping, 1981. (With contributions from **Hein** and **Hägglund.**)

117. **Kristo Ivanov:** Forskningsanknytning av Universitetets grundutbildning. LiU-MAT-ADB-R-80-1

118. **Kristo Ivanov:** Systemvetenskap och fragmentering av kunskap. LiU-MAT-ADB-R-80-2

119. **Kristo Ivanov:** Mot ett ingenjörsvetenskapligt universitet. LiU-IDA-R-84-2

120. **Kristo Ivanov:** Några policy-riktlinjer för ämnet ADB. LiU-MAT-R-83-3

121. **H. Jan Komorowski** (Ed.): Proceedings of the Symposium on Prolog Programming Environments, Linköping, 1982.

122. **Bengt Lennartsson:** Programvarumiljöer. Produktionsteknik för programvara i Ada och andra språk. LiTH-IDA-R-84-01

123. **Hans Lunell:** Tre skisser om datalogi som vetenskap LiTH-MAT-R-81-16

124. **Erik Sandewall:** Datavetenskaplig utvecklingsmiljö och kunskapsöverföringsprogram. LiTH-IDA-R-83-10

125. **Dan Strömberg:** Datorn - hjälpreda eller hot i det lilla företaget? (LiTH-MAT-R-81-06)

126. **Dan Strömberg:** Gränserna för artificiell intelligence - en reseskildring (LiTH-MAT-R-82-46)

127. **Dan Strömberg:** Ett kritiskt perspektiv på artificiell intelligens forskning. LiU-MAT-R-82-1

128. **Bo Sundgren:** Conceptual Design of Databases and Information Systems. LiTH-IDA-R-84-09 (Lecture Notes)

Systems Documentations 1980-84:

129. MEDICS - Systemdokumentation och användarhandledning. **Hans Holmgren, Sture Hägglund, Olle Rosin.** (SSRC Systemdok. 17)

130. MEDICS - Författarmanual - Preliminär version. **Hans Holmgren, Sture Hägglund, Olle Rosin.** (SSRC Systemdok. 18)

131. MINISCOPE - Användarhandledning. **Lars Wikstrand.** (SSRC Systemdok. 19)

132. IDECS3 Reference Manual. **Sture Hägglund.** (SSRC Systemdok. 20)

133. ED3 - User's Guide **Ola Strömfors.** (SSRC Systemdok. 21)

134. SCREBAS - Provisional Reference Manual. **Erik Sandewall.** (SSRC Systemdok. 22)

135. Ett gränssnitt mellan LISP 1.6 och MIMER på DEC-10 vid LIDAC. Användarhandledning. **Hans Holmgren.** (SSRC Systemdok. 23)

136. ALGOL68C - Release 1.271, Users Guide. **Arne Fäldt.** (SSRC Systemdok. 24)

137. Handledning i användande av PIG. **Olle Willen.** (SSRC Systemdok. 25)

138. The LOIS Manager. **Erik Sandewall.** (SSRC Systemdok. 26)

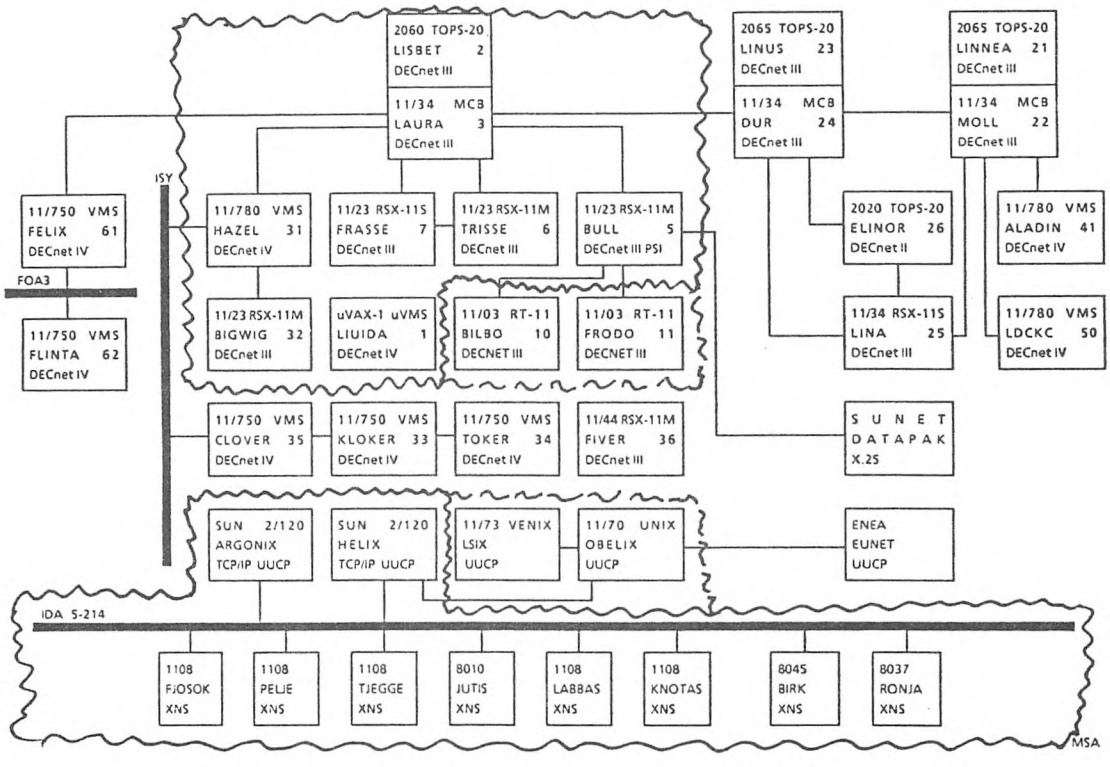
139. AFORM User's Guide, **Arne Fäldt.** (SSRC Systemdok. 27.1)

Appendix 5.

Computer Facilities.

The department has a policy of giving high priority to providing ample computing resources for research and education. We have also during the years been able to modernize and keep in pace with the rapid development in the area, e.g. regarding the emergence of powerful workstations with high-resolution graphics and high-performance CPU. Our orientation towards experimental computer science makes such a policy especially important and we believe that adequate computer equipment is essential for the quality of research and education.

Our main computer resources for research are a DECsystem-2060 (there is an additional system for undergraduate education), a VAX 780 (which is shared with the Physics department) and a Xerox Ethernet with six 1108 Lisp Machines, file server and laser printer. We have also recently acquired a few SUN workstations. In addition there are lots of smaller computers (MicroVax, PDP-11:s, MacIntosh and other PC:s of various kinds.) There are also special purpose equipment, especially for text processing.



1. ***Computer network at the university, with the department's research machines circled.***

Universitetsområdet Valla