# Activity and Status Report

# The Software Development Environments Group

# September 1999

Department of Computer Science
Lund Institute of Technology
Lund University
Sweden

LUNDS TEKNISKA HÖGSKOLA
Lunds universitet

## Members


| | |
|---|---|
| Boris Magnusson, Prof. | Boris.Magnusson@cs.lth.se |
| Görel Hedin, Ph.D. | Gorel.Hedin@cs.lth.se |
| Roger Henriksson, Ph.D. | Roger.Henriksson@cs.lth.se |
| Klas Nilsson, Ph.D. | Klas.Nilsson@cs.lth.se |
| Jonas Skeppstedt, Ph.D. | Jonas.Skeppstedt@cs.lth.se |
| | |
| Ulf Asklund, Ph.Lic. | Ulf.Asklund@cs.lth.se |
| Daniel Einarsson, Ph.Lic. | Daniel.Einarsson@cs.lth.se |
| Anders Ive, M. Sc. | Anders.Ive@cs.lth.se |
| Patrik Persson, M. Sc. | Patrik.Persson@cs.lth.se |
| Mathias Haage, B. Sc. | Mathias.Haage@cs.lth.se |
| Eva Magnusson, B. Sc. | Eva.Magnusson@cs.lth.se |
| Anders Nilsson, M. Sc. | Anders.Nilsson@cs.lth.se |
| Jonas Persson, M. Sc. | Jonas.Persson@cs.lth.se |
| Sven Gestegård Robertz, M. Sc. | Sven.Gestegard@cs.lth.se |


## Web site

http://www.cs.lth.se/Research/ProgEnv/

## Address


Department of Computer Science
Lund University
Box 118
SE-221 00 Lund, Sweden


## Telephone


+46-46-222 00 00 (Lund University)


## Fax


+46-46-13-10 21 (Dept. fax)


## CONTACT US!

Contact us if you are interested in more information, research cooperation, master's thesis projects, doctoral studies, etc.

# 1 INTRODUCTION

The Software Development Environments Group at the Department of Computer Science at Lund Institute of Technology, has been active since the early 1980's. The research is focused on integrated software development environments, in particular for object-oriented languages. Areas of speciality include configuration management, language design, language implementation, real-time computing, robotics, and code generation for multiprocessors.

The overall aim of our research is to develop techniques for making software production more efficient. We pursue this goal by identifying key techniques that are limiting factors, by developing enabling techniques when necessary, and by building environments to demonstrate and evaluate the developed techniques. The research is centered around a number of core areas within software development support, with central themes of integrated environments, object-oriented languages (in the tradition of Simula, BETA, and Java), and embedded systems such as industrial robots and mobile phones. The research method is focused on experimental implementation and development of theory that is of practical relevance. There is much interaction between the individual projects, with a strive for integrating the developed core technologies. Many of the projects are done in collaboration with industry. See the back cover for an overview.

# 2 CONFIGURATION MANAGEMENT

Within the area of configuration management our focus is on support for fine-grained integrated configuration management with optimistic development and collaborative awareness. A major problem in large-scale software development is to support collaboration. The problem is particularly evident in large development teams geographically distributed in the country or in the world. By *collaborative software development environments* we mean environments which *explicitly* support the collaboration between team members in the task of developing software systems. This includes mechanisms for providing support for sharing software and documents, awareness about activities in the team, and coordination between different tasks performed by different team members. It also includes support in different activities such as design, implementation, testing, maintenance, and documentation.

## 2.1 Collaborative CM: The COOP/Orm model and system

*Boris Magnusson, Ulf Asklund, Jonas Persson*

To support collaborative software development we are developing a model for semi-synchronous collaborative editing, spanning the gap between asynchronous and synchronous editing styles. The model is based on hierarchically partitioned documents, fine-grained extensional version control and configuration management [11], and a mechanism we call *active diffs* for supplying collaboration awareness. Furthermore, the model supports optimistic check-out with merge support and change-oriented representation of differences. It is designed to work also in a distributed wide area network. It supports replication of data at several sites with synchronization after network failures. The implementation, COOP/Orm, currently supports the semi-synchronous editing model for hierarchical text documents. See Figure 1.

## 2.2 NUTEK project: Collaborative Systems Development

*Boris Magnusson, Ulf Asklund, Jonas Persson*

Producing and maintaining software is a growing activity, and problem, for Swedish industry. In this project we focus on teamware, i.e., support for groups of developers that work together in tasks such as designing, implementing and documenting software systems. This is an increasing problem as development groups grow and become more geographically dis-

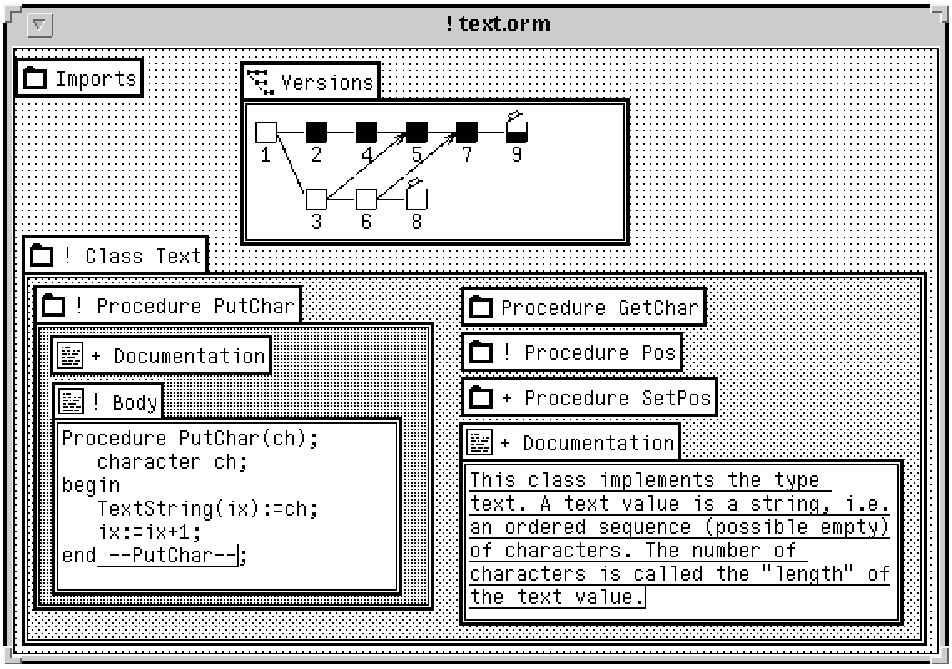


**Figure 1** *The COOP/Orm system. The figure shows the creation of a new version (9) of a document. A comparison to an older version (2) is shown as the text is being edited, e.g. by underlining text that is new in (9) compared to (2). Some earlier versions have been created by merging two versions, e.g. version (7) is a merge of (5) and (6). Another user is simultaneously working on the document (creating version 8), so the user is aware that a future merge between 8 and 9 will probably be necessary.*

tributed. We work with solutions which build on a fine-grained revision control system, optimistic check-out, and advanced support for merging, see 2.1 (Collaborative CM: The COOP/Orm model and system).

We also participate in the group on Configuration Management organized by VI (Sveriges Verkstadsindustrier). VI has participants from leading Swedish companies and works as a bridge between the companies and to the research world. As part of this cooperation, we are investigating current practice in Swedish industry, concerning configuration management [2, 12]. Our prototype system, COOP/Orm, will be evaluated in an industrial setting in this project. In addition to VI, we cooperate with the following companies:

- Ericsson Mobile Communications
- Kockums Computer Systems
- Ericsson Microwave

# 3 LANGUAGE DESIGN

Within the area of language design our focus is on object-oriented domain-specific languages and the use of interactive environments for evolving languages. A number of individual projects are pursued in this direction, most of them making use of the APPLAB tool for interactive language development described in 4.1.

## 3.1 Robot languages

*Görel Hedin, Klas Nilsson*

We are interested in domain-specific languages for robot programming, the programming of robots at different levels, and the integration of on-line and off-line programming. We have implemented the robot language RAPID and experimented with new language constructs to better support various robot programming problems. As a language implementation tool we have used the APPLAB system. The largest case study so far is the implementation of RAPID to control an ABB Irb-6 industrial robot [5, 13]. This research is done in cooperation with ABB.

### 3.2 Language support for design patterns and frameworks

*Görel Hedin, Eva Magnusson*
We are interested in how different language constructs support design patterns and frameworks, and how the intentions of design patterns and frameworks can be captured either directly by language constructs or by annotating programs with extra information [15]. We use the APPLAB system as an experimental platform for trying out the ideas. Within the project we cooperate with the Department of Computer Science at Aarhus University, Denmark.

### 3.3 NUTEK project: Executable Specifications for Interactive Communicating Devices

*Sven Gestegård, Görel Hedin, Boris Magnusson*
This project is initiated by Ericsson Mobile Communications to support development of software for interactive communicating devices. New radio technology like BlueTooth for short range cordless communication enables new kinds of distributed applications involving mobile devices that communicate in a dynamic way. Current practice for developing software for these kinds of applications is to stepwise transform informal specifications into implementations. This has the drawback that problems are revealed late in the development process, during integration and validation.

In this project we develop a new technique based on a combination of a customized executable specification language, an interactive support environment, and a simulated technical environment. The goal is to enable detection of design problems early in the development process. We use the APPLAB system as an experimental platform for developing the new specification language and its implementation. The specification language is a statechart-like language, but specifically customized to the Bluetooth communication protocol.

The project is done in cooperation with Ericsson Mobile Comunications.

### 3.4 Object-Oriented Product Modeling

*Görel Hedin*
We investigate models, languages, and interactive tool support for product configuration. A product modelling language, based on object-oriented grammars, has been developed and applied to an example problem: mechanical configuration of plate heat exchangers [21]. We cooperate with Alfa Laval Thermal within this project.

## 4 LANGUAGE IMPLEMENTATION

Within the area of language implementation our focus is on safe object-oriented languages, integrated and incremental language-based editing environments, object-oriented language specification, and small target environments.

## 4.1 The APPLAB system

*Görel Hedin*

We are interested in supporting the fast and easy development of new languages, in particular application languages (or domain-specific languages). As a research platform we have developed APPLAB (APPlication language LABoratory), an interactive meta-environment for the development of domain-specific languages and programming environments for these languages [5, 13, 16]. APPLAB is based on syntax-directed editing, semantics-directed editing, incremental parsing, and reference attributed grammars for semantic checking and code generation. Figure 2 shows the use of APPLAB to predict timing and memory usage in real-time computing, as described in 5.1 (ARTES project: Integrated Control and Scheduling). Key properties of APPLAB include that the language grammars can be edited at the same time as an example program in the new language, and that a language-based environment is available at all times for both the grammars and example program.

## 4.2 Object-oriented attribute grammars

*Görel Hedin*

We are interested in being able to easily specify the static semantics of languages, to allow fast and compact evaluators to be generated from such specifications, to support incremental evaluation, and to use the static semantics information in language-based editing systems. In particular, we are interested in supporting object-oriented languages that have fairly complex non-local dependencies resulting from language constructs such as inheritance and qualified access. Coming from a tradition of object-oriented programming it has also been natural to design the specification languages in an object-oriented way, e.g. viewing nonterminals and productions as abstract and concrete classes over syntax tree node objects, and to support attributes that are references to other nodes in the syntax tree.

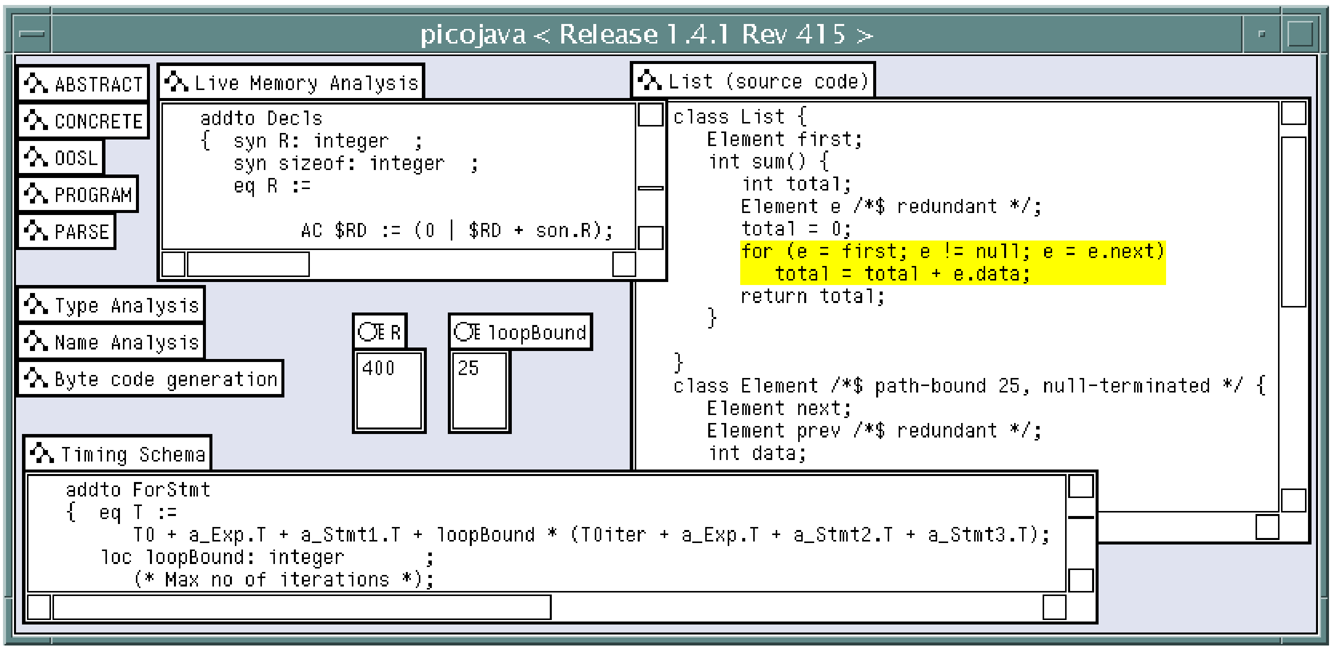


**Figure 2** *Example timing and memory predictions in APPLAB. The* List *window to the right shows the source code of the analyzed program. The window labelled* R *shows the maximum amount of memory possibly referenced from the* first *reference in the* List *class, and the window labelled* loopBound *shows the maximal number of iterations in the for loop. The computation of* R *and* loopBound *are specified in the* Live Memory Analysis *and* Timing Schema *windows, respectively.*

One developed technology is *Door Attribute Grammars* that supports declarative specification of attributions containing objects and references. The resulting attributions can be efficiently updated incrementally, and have been successfully applied to describe and implement incremental static-semantics for object-oriented languages [7]. Another developed technology is *Reference Attributed Grammars* that is similar to Door AGs, but allows fully automatic evaluation (although not incremental evaluation so far) [16]. Reference Attributed Grammars are used in the APPLAB system.

### 4.3 Static semantics of object-oriented languages

*Eva Magnusson, Görel Hedin*
We are interested in object-oriented language constructs and how they can be specified in Reference Attributed Grammars. We use the APPLAB system as an experimental platform for trying out the specifications. Currently, we are investigating the key constructs in Java, namely: single inheritance for classes, multiple inheritance for interfaces, method overloading, qualified access, virtual methods, block structure in combination with inheritance (inner classes), and assignment compatibility in connection to subtyping [16].

For future work, we are interested in analyzing constructs of other languages, e.g. various approaches to generics, such as virtual classes in BETA.

### 4.4 Real-time garbage collection

*Roger Henriksson, Boris Magnusson*
This project is focused on dynamic memory management in embedded real-time systems. In real-time programming, programs usually run for a long period of time, and proper memory management becomes a crucial issue in order to avoid memory fragmentation and memory exhaustion. Memory management programmed into the application (using 'free' or 'dispose' calls) have turned out to be very error-prone since both dangling pointers and memory leaks must be avoided. Automatic memory management sounds ideal, but it is, however, not trivial to integrate in these systems since parts of the system are usually executing under hard real-time requirements, not allowing arbitrary pauses for garbage collection.

A new garbage collection scheduling algorithm has been developed suitable for applications with *hard* real-time constraints, such as control systems (embedded systems) [4]. The scheduling strategy employs properties of control systems to ensure that no garbage collection work is performed during the execution of critical processes. The algorithm has been implemented in an industrially relevant real-time environment in order to show that the strategy is feasible in practice, giving sub-millisecond worst-case delays on standard hardware, even in a non-optimized prototype garbage collector implementation. See Figure 3. We cooperate with the department of Automatic Control, LTH, within this project.

### 4.5 NUTEK project: Dynamic Distributed Software Supporting Tiny Mobile Systems

*Anders Ive, Roger Henriksson, Boris Magnusson, Klas Nilsson*
Telecommunications systems and manufacturing systems are complex systems of key interest for Swedish industry. In both areas, the use of web-technologies such as plug-ins, Java, and applets, have received much attention within industry. However, in two specific and important applications, cellular phones and industrial robots, severe demands on memory usage and real-time properties pose special problems. A number of trade-offs have to be investigated and verified by prototype implementations.

As part of this project we are implementing a small memory footprint JVM. The research is done in cooperation with Ericsson Mobile Communications and ABB Robotics Products.

### 4.6 Java-to-C translator

*Anders Nilsson, Klas Nilsson*
We develop a Java-to-C translator. The intended use is to program small embedded systems in Java and to have control over the resulting C-code to adapt it to the embedded system. This project is part of the NUTEK project Open Control Architectures. The implementation is done using the APPLAB system.

## 5 REALTIME COMPUTING

Within the area of realtime computing our focus is on support for hard real-time in object-oriented languages.

### 5.1 ARTES project: Integrated Control and Scheduling

*Patrik Persson, Görel Hedin, Klas Nilsson, Boris Magnusson*
This is a joint project with Department of Automatic Control, LTH. In our part of the project, the focus is on using attribute grammars and incremental semantic analysis to carry out on-line interactive analysis of the worst-case timing of object-oriented software and generation of exception handling code for coping with unexpected delays. We also analyze memory requirements for integration with real-time garbage collection [18]. The APPLAB system is used as an experimental platform, see Figure 2. In addition to Dept. of Automatic Control, LTH, we cooperate with Sigma Exallon Systems.

## 6 ROBOTICS AND AUTOMATION

The emphasis within this area is on open software architectures for industrial robots.

### 6.1 Experimental lab for industrial robots

*Klas Nilsson*
The laboratory for robotics and real-time systems is centered around an ABB Irb-6 robot and an ABB Irb-2000 robot (see Figure 3). Hardware interfaces have been developed to create an open system suitable for control experiments. The computer hardware is VME-based with both micro processors and signal processors integrated into an embedded system for hard real-time control. The system is connected to a network with Sun workstations, which are used for program development and control design. A purpose of the current project is to show how to organize open robot control systems and to verify these ideas by means of experiments.

The work is done in close cooperation with the Department of Automatic Control, where the robot lab is hosted.

### 6.2 Program and robot visualization

*Mathias Haage, Klas Nilsson*
The use and implementation of graphics for embedded programming interfaces is explored [14]. We have implemented an integrated robot simulation environment using vrml, eai, and Java.

## 6.3 Hierarchical concurrent systems

*Daniel Einarsson, Klas Nilsson*

We investigate language constructs and runtime support for enabling programming of hierarchical concurrent systems, both for control and simulation [3].

## 6.4 NUTEK project: Open Control Architectures

*Klas Nilsson*

Manufacturing systems are complex systems including control and programming of individual machines, production cells, and complete manufacturing systems. A key issue for Swedish industry is to achieve high productivity in such systems. Technically, this means that both efficiency and flexibility should be accomplished on different levels of the system. Together with our industrial partners, ABB Robotics Products and ABB Corporate Research, we have investigated the need for open modular controllers, which deserves a dedicated research effort. This project will involve collaboration with other disciplines such as Automatic Control, but the main focus is on the software aspects [17, 22].

Subprojects of this project include 6.1 (Experimental lab for industrial robots), 6.2 (Program and robot visualization), 6.3 (Hierarchical concurrent systems), and 4.6 (Java-to-C translator).

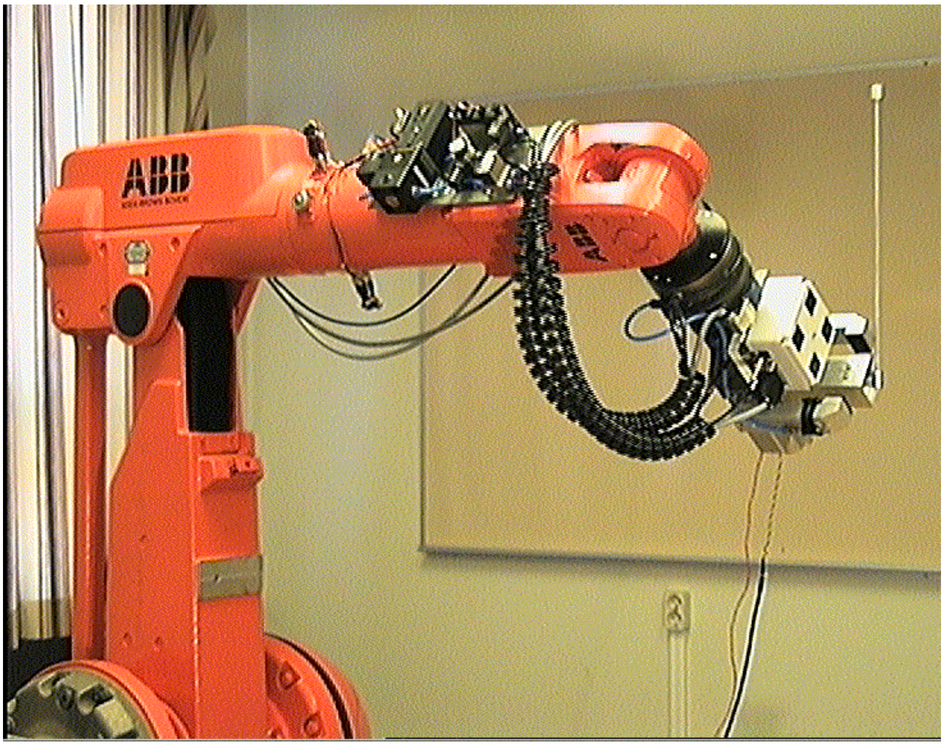


**Figure 3** *An ABB IRB-2000 industrial robot balances an inverted pendulum. The tasks in the control program make use of dynamically allocated memory that is garbage collected in real-time. Sub-millisecond response times for the critical processes are guaranteed by the real-time GC algorithm.*

# 7 CODE GENERATION FOR MULTIPROCESSORS

The goal of this research is to develop and evaluate compiler algorithms to reduce the performance penalty due to cache misses in multiprocessors. The current emphasis is on data prefetching.

## 7.1 Data prefetching

*Jonas Skeppstedt*

Data prefetching is a promising approach to reduce the performance penalty due to cache misses, especially for multiprocessors. Traditionally, compiler-controlled prefetching inserts special prefetch instructions in loops in order to fetch data before it is needed by the processor. While this approach removes cache misses, it can also increase the instruction count, sometimes significantly. We investigate new prefetch approaches which provide the same benefits of a reduced number of cache misses, while producing a minimal increase in instruction count [19, 20]. This is work done in cooperation with Professor Michel Dubois at the University of Southern California, Los Angeles.

# PUBLICATIONS

## Books


[1] J. Lindskov Knudsen, M. Löfgren, O. Lehrmann Madsen, B. Magnusson. *Object-Oriented Environments: The Mjølner Approach*. Prentice-Hall, 1993.


## Theses


[2] U. Asklund. *Configuration Management for Distributed Development - Practice and Needs*. Licentiate thesis, Lund Institute of Technology, June 1999.

[3] D. Einarson. *Model-Oriented Programming*. Licentiate thesis, Lund University, February, 1999.

[4] R. Henriksson. *Scheduling Garbage Collection in Embedded Systems*. Ph.D. thesis, Lund Institute of Technology, September 1998.

[5] E. Bjarnason. *Interactive Tool Support for Domain-Specific Languages*. Licentiate thesis, Lund University, December 1997.

[6] R. Henriksson. *Scheduling Real-Time Garbage Collection*. Licentiate thesis, Lund Institute of Technology, January 1996.

[7] G. Hedin. *Incremental Semantic Analysis*. Ph.D. thesis, Lund Institute of Technology, 1992.

[8] S. Minör. *On Structure-Oriented Editing*. Ph.D. thesis, Lund Institute of Technology, 1990.

[9] M. Bengtsson. *Real-Time Compacting Garbage Collection Algorithms*. Licentiate thesis, Lund Institute of Technology, 1990.

[10] A. Gustavsson. *Software Configuration Management in an Integrated Environment*. Licentiate thesis, Lund University, 1990.

## Selected recent papers


[11] U. Asklund, L. Bendix, H. Christensen, and B. Magnusson. The Unified Extensional Versioning Model. In *Proceedings of SCM-9, Ninth International Symposium on System Configuration Management*, J. Estublier (Ed.), Toulouse, France, September 1999. LNCS, Springer Verlag.

[12] U. Asklund, B. Magnusson, and A. Persson. Experiences; Distributed Development and Software Configuration Management. In *Proceedings of SCM-9, Ninth International Symposium on System Configuration Management*, J. Estublier (Ed.), Toulouse, France, September 1999. LNCS, Springer Verlag.

[13] E. Bjarnason, G. Hedin, K. Nilsson. Interactive Language Development for Embedded Systems. In *Nordic Journal of Computing* 6(1999), 36-55.

[14] M. Haage, K. Nilsson, On the Scalability of Visualization in Manufacturing. To appear in IEEE *ETFA'99 (Emerging Technologies in Factory Automation)*, 1999.

[15] G. Hedin, J. L. Knudsen. Language Support for Application Framework Design. In *Implementing Application Frameworks: Object-Oriented Frameworks at Work*, M. Fayad, D. Schmidt, R. Johnson (Eds.), Wiley, ISBN 0471252018, 1999.

[16] G. Hedin. Reference Attributed Grammars. To appear in *Informatica*.

[17] K. Nilsson, R. Johansson. Integrated Architecture for Industrial Robot Programming and Control. Accepted for publication in *Robotics and Autonomous Systems*, Elsevier Science.

[18] P. Persson. Live Memory Analysis for Garbage Collection in Embedded Systems. In Proceedings of the *ACM SIGPLAN 1999 Workshop on Languages, Compilers, and Tools for Embedded Systems* (LCTES'99), Atlanta, GA, May 1999, pages 45-54. ACM SIGPLAN Notices, July 1999.

[19] J. Skeppstedt, F. Dahlgren, P. Stenström. Evaluation of Compiler-Controlled Updating to Reduce Coherence-Miss Penalties in Shared-Memory Multiprocessors. *Journal of Parallel and Distributed Computing*, v 56, n 1, February, 1999, p122-143, 1999, Academic Press.

[20] J. Skeppstedt, M. Dubois. Compiler Controlled Prefetching for Multiprocessors Using Low-Overhead Traps and Prefetch Engines. *Journal of Parallel and Distributed Computing*, in press.

[21] G. Hedin, L. Ohlsson, J. McKenna. Product Configuration Using Object-Oriented Grammars. In *Proceedings of the 8th International Symposium on System Configuration Management* (SCM-8), Brussels, July 20-21, 1998. LNCS 1439, pp 107-126, Springer-Verlag.

[22] K. Nilsson, A. Blomdell, O. Laurin. Open Embedded Control. In *Real-Time Systems*, Vol 14, No 3, pp 325-343, Kluwer, May 1998.

# *Research at the Software Developments Environments Group, LTH*