Design Guidelines and Visualization Support for Developing Parallel Real-Time Applications, P4-9805

Lars Lundberg
Department of Software Engineering and Computer Science, University of Karlskrona/Ronneby,
Soft Center, S-372 25 Ronneby, Sweden, Phone: +46-(0)457-385833, Fax: +46-(0)457-271 25
Lars.Lundberg@ipd.hk-r.se

Proposal for a 1 year continuation of a research project within PAMP

1. Short summary

This is a continuation of a project supporting two Ph.D. students: Magnus Broberg and Daniel Häggander. Magnus and Daniel have received money for two years. Their projects are progressing nicely (I have attached an updated version of the activity report for 1999, including a letter from our industrial partner Ericsson Software Technology). Magnus has written or coauthored five papers: three published in international conferences, one in a national workshop, and one in second review for publication in a journal. Daniel has written or coauthored eight papers: six in international conferences, one in a national workshop, and one in first review for journal publication. Daniel is also in the process of sending in a patent application. Magnus and Daniel both took their licentiate degrees during 1999, and I expect that they will graduate in early 2002. I would therefore need one additional year of funding for each of them. Magnus and Daniel started their Ph.D. studies somewhat before we got the funding from ARTES/PAMP. This is the reason why we only need funding for one additional year. ARTES allocates a fixed amount for each Ph.D. student (600 kkr/year), i.e., we apply for 2 x 600 kkr = 1.2 Mkr.

2. Problem statement

Many real-time applications, especially in the telecommunication domain, are *transaction oriented*. Failure to meet the performance requirements, e.g. due to transient overloads, often results in loss of value for the company or organization operating the real-time system, i.e. *the quality of the service provided by the real-time system to the company operating it may seriously deteriorate if the timing requirements are not met*. One example is that telephone network operators may lose information about billable calls, resulting in loss of money. The real-time demands on transaction oriented applications are increasing, e.g. there is a need for including *hot billing* in the billing systems. This means that the cost for making a phone call is calculated in real-time, making it possible to disconnect subscribers which have exceeded their cost limit. Consequently, there is a need for *high and predictable performance*.

Due to the rapid growth of the telecommunication sector, most real-time systems are expected to scale up in the near future, e.g. network operators want to be able to handle more subscribers. Due to their size and complexity, many real-time telecommunication systems are extremely costly to develop and maintain. Consequently, one would like to increase the capacity of such systems in a modular fashion (*scalable performance*), i.e. one would not like to redesign large parts of the system when the performance requirements increase.

Today multiprocessors that share the same address space, so called *symmetric multiprocessor systems* (SMPs), are being widely used in industry. Besides the performance and cost advantages of this emerging technology, one can (at least in theory) incrementally increase throughput and decrease response time by simply adding more processors in a modular fashion. In order to reduce the cost for developing and maintaining large real-time systems, one is often forced to use standard system software, such as standard operating and database systems. Consequently, the SMP platform consists not only of the multiprocessor hardware; it also includes system software.

In order to benefit from multiprocessors, the application program must be structured in a parallel way. One way of doing this is to write multithreaded programs using threads. Today, the experience of writing parallel applications is limited, especially if high, predictable and scalable performance is the goal. Moreover, there can be a conflict between maintainable and reusable software design on the one hand and high, predictable and scalable performance on the other hand [8][9].

Consequently, there is a need for tools and guidelines for developing parallel real-time systems with high, predictable, and scalable performance. Due to the considerable costs for developing and maintaining large systems, there is also a need for understanding the trade-offs between performance and reusability/maintainability.

3. Main ideas

In this project we will extend a prototype [2] of a visualization and performance prediction tool for parallel applications written using Posix threads. The tool consists of three major parts, the Recorder, the Simulator, and the Visualizer (see Figure 1). The developer writes the multithreaded program, compiles it, and an executable binary file is obtained. After that, the program is executed on a uni-processor, with the *Recorder* placed *between* the program and the standard thread library. Each time the program uses the thread routines, the call passes through the *Recorder* which records information about the call. The *Recorder* then calls the original routine in the thread library. When the execution of the program finishes all the collected information is stored in a file, the *recorded information*.

The *Simulator* simulates a multiprocessor execution; we currently support Sun Solaris and Linux. The main input for the simulator is the *recorded information*. The simulator also takes as input the hardware configuration and scheduling policies. The output from the simulator is information describing the simulated execution. Using the *Visualizer* the simulated parallel execution of the program can be inspected. When visualizing a simulated execution, it is possible for the developer to click on a certain interesting event, get the source code displayed, and the line making the call that generated the highlighted event.

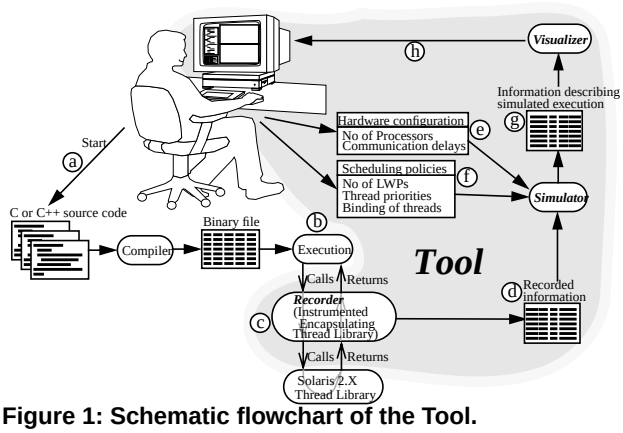

Figure 1: Schematic flowchart of the Tool.

The experience we have from the tool is promising. We would however, like to include two extensions:

- Mechanism for handling recordings from a certain period of the execution. Today, we assume that we monitor the parallel program from start to completion. This is not practical for some long-running programs, e.g. some real-time applications where the same program is executed endlessly over and over again.
- Using the tool for obtaining dynamic priority schemes that will minimize the execution time. The tool can detect the critical path in a parallel program executing on a multiprocessor, and by increasing the priority of the segments that are in the critical path we expect to be able to improve the performance.

In order to obtain the guidelines for developing parallel programs with high, predictable and scalable performance we have performed a number of case studies using industrial parallel real-time applications. These case studies show that there is a conflict between maintainability and SMP performance. The main reason for this conflict is that object oriented design techniques which are aimed at maximum maintainability tend to generate excessive dynamic memory accesses. Based on our experiences from the case studies we have developed a set of design guidelines that make it possible to balance maintainability and performance.

In the next and final phase of the project we are going to apply our guidelines to a real development project at Ericsson and evaluate the results in terms of performance and (estimated) maintainability. We are also going to develop efficient memory management techniques which make it possible to obtain high performance even when there is a lot of dynamic memory accesses.

The assessment of the impact that our guidelines have on maintainability and reusability will be done together with researchers from the ARCS research group at Karlskrona/Ronneby. This group is headed by professor Jan Bosch and the group has been working on reusability and maintainability aspects for a number of years.

Finally, our main research vehicle is an eight processor SMP that we will use as a test bench to do performance measurements, as a platform to run simulations on, and as platform for parallel program development.

4. Expected results and impact

For Daniel's part of the project we expect the following results:

- We expect to validate, and possible modify, our set of guidelines on how to design parallel real-time applications with high, predictable and scalable performance. The guidelines also consider reusability and maintainability aspects.
- An efficient dynamic memory management algorithm that, using compile time information, will make it to obtain high performance even when there is a lot of dynamic memory accesses.
- Ph.D. thesis for Daniel.

For Magnus' part of the project we expect the following results:

- An extension of the existing tool, which makes it possible to handling recordings from a certain period of the execution.
- A technique which makes it possible to obtain and implement dynamic priority assignment schemes that will minimize the execution time of a parallel program on an SMP.
- Ph.D. thesis for Magnus.

5. Project plan

Two Ph.D. students - Magnus Broberg and Daniel Häggander - will perform most of the work in this project. Magnus and Daniel have already done most of their teaching for their period as Ph.D. students. This means that I expect that the current funding from PAMP/ARTES for Magnus and Daniel will end in March 2001. I expect that they will complete their Ph.Ds in March 2002 and work with the project full time the last year, i.e. we will need funding for one year (2001-04-01 -- 2002-03-31). We present individual plans for Magnus and Daniel:

Magnus Broberg (performance prediction and visualization tool):

Activity 010401-010930: *Extend the existing tool with techniques that make it possible to handle recordings from a certain period of the execution.*
In the current version of our tool we can only handle recording from a complete program execution, i.e. recordings from the start of the program to the program terminates. In many cases one would like to monitor certain periods of the program's execution. This is particularly interesting for real time embedded systems where the same program is executed endlessly over and over again. Another reason for restricting the monitoring to certain periods is that we would like to limit the amount of recorded information. The major problem with this extension is that we do not know the state of the semaphores and mutex variables etc. when we turn on our recordings. We will implement and evaluate a number of alternative approaches, e.g. simply assuming some state, or trying to keep track of the state during the periods when no values are recorded. Ericsson will provide case applications for the evaluation process.
Milestone: *A report describing our approach to handle incomplete recordings.*

Activity 011001-020331: *Develop techniques for calculating and implementing dynamic priority assignment schemes based on critical path analysis.*
Our tool can identify the critical path in a parallel program executing on a multiprocessor (in general the critical path depends on the number of processors). The main purpose for doing a critical path analysis is that we want to identify the routines that one should concentrate the optimization efforts on when using a multiprocessor with *n* processors. however, the critical path analysis can also be used for introducing a dynamic priority scheme that will minimize the execution time on a multiprocessor. We will implement and evaluate this technique. Ericsson will provide a case applications for the evaluation process.
Milestone: *Ph.D. degree for Magnus*

Daniel Häggander (Guidelines):
The two activities below may, to some extent, overlap in time and thus be carried out in parallel.

Activity 010401-010930: *Evaluate the previously developed guidelines on a real development project.*
By looking at three existing large applications, we have developed a set of guidelines and a simple process for how to use these guidelines. We have, however, not applied the guidelines to any real development project. We will now apply the guidelines to a real development project at Ericsson. It will probably be a challenging application for handling so called pre-paid subscribers, i.e. subscribers that have to pay in advance. Each such subscriber has an account that must be decremented in real-time during the call. If the account becomes empty the call is terminated.
Milestone: *Report on the evaluation.*

Activity 011001-020331: *Develop a technique for obtaining efficient dynamic memory management.*
Our experiences show that the ambition of writing maintainable programs causes a lot of dynamic memory allocations and deallocations. These allocations and deallocations can turn into a serious performance bottle-neck when using an SMP. We have some initial ideas on how to solve these problems using compile time information, thus making it possible to develop the programs without considering the dynamic memory problem. This will make it possible to achieve good maintainability without sacrificing performance. If our approach is as good as we think, we plan to protect it with a patent.
Milestone: *Ph.D. degree for Daniel.*

6. Budget

ARTES allocates a fixed amount for each Ph.D. student (600 kkr/year) and we will manage on that, i.e., we apply for 2 x 600 kkr = 1.2 Mkr.

7. Related research

There are guidelines for designing software with high performance, e.g. the “Principles for creating responsive Software” in [16]. However, these guidelines consider neither the specific problems of parallel programs and multiprocessor systems nor maintainability and reusability aspects. Guidelines for designing maintainable and reusable software often take the form of design patterns, i.e. examples of best practice solutions [5]. In most cases, such design patterns do not consider the trade-offs between performance and maintainability/reusability. Experience shows that parallel real-time applications developed according to such guidelines may suffer from poor performance [8][9].

The Promis generic database [4] is one example of a commercial product for which it is possible to systematically calculate the performance degradation when increasing the maintainability and flexibility of the application.

We are not the only ones that have realized that there is a need for doing trade-offs between different architectural aspects, e.g. performance versus maintainability/reusability. The ATA project (Architectural Tradeoff Analysis method) at Carnegie Mellon University also attacks this problem (see http://www.sei.cmu.edu/activities/ata/ata_init.html). However, their approach do not focus on parallel real-time systems.

Many efficient dynamic memory allocators have been designed. Wilson, Johnstone, Neely and Boles have written an excellent survey [18]. The main focus has been sequential allocators, with the emphasis on speed and memory utilization. Benjamin Zorn maintains a site with links to dynamic memory allocators that are available on the net [19]. The allocator designed by Dough Lea has been found to be both the fastest and the most efficient on several applications [13]. Gloger has created a scalable version of this allocator - ptmalloc [6]. Another academic memory allocator for SMPs is Hoard [1]. There is also a commercial SMP version of SmartHeap [14]. We expect that our optimization of dynamic memory will outperform these approaches because we will benefit from compile time information about the application program.

Different tools for visualizing the behaviour of, and thus the bottlenecks in, parallel programs have been developed [3, 7, 10, 11, 12, 15, 17]. Some performance tools show the behaviour of one particular monitored multiprocessor execution of the parallel program [3, 7, 10, 11]. If we monitor the execution on a multiprocessor with four processors such tools make it possible to detect bottlenecks which are present when using four processors. The problem with this approach is that one cannot detect bottlenecks which appear when using another number of processors. There are a number of tools which make it possible to visualize the (predicted) behaviour of a parallel program using any number of processors. However, these tools are either developed for message passing systems [12] or for non-standard programming environments [14, 16], thus making it impossible to use them for industrial applications.

8. Relation to the profile

The objective of PAMP is to provide design methods to make it possible to use symmetric multiprocessor technology to meet the performance demands of emerging high-performance real-time applications. PAMP is intended to develop systems design methods that shall drastically shorten the development time. Examples of research issues within the PAMP profile are: methods for parallelization of applications, and methods for performance predictions of multiprocessor systems.

The guidelines and tool produced in this project fit the PAMP profile. The tool will increase the productivity of the developer as well as the performance of the parallel real-time system. The guidelines will make it possible to parallelize an application in order to obtain high, predictable and scalable performance. The guidelines will also take the development time into consideration, since we will consider maintainability and reusability as well as performance.

9. Industrial relevance

The experience of writing parallel real-time applications is very limited an there is a need for guidelines and tools that will help the developer. Examples of questions that have been asked by our industrial partner Ericsson is:

- "How fast will the application run if the number of processors is increased?"
- "How can the application be restructured in order to achieve better speedup?"
- "What guidelines should be followed when writing parallel applications?"

The first two questions could be handled by the performance prediction and visualization tool, and the guidelines developed in this project are a direct answer to the last question. It is of particular industrial relevance that our guidelines will not only consider high capacity; they will also consider the important issues of system development cost (maintainability and reusability).

10. Relation to other SSF programmes

The Personal Computing and Communication (PCC) programme aims at providing "Mobile multimedia services to all at the same price level as fixed telephony today". This is obviously a challenging goal that will require high performance processing. One of the most promising ways to obtain high performance is parallel processing using SMPs, which is being studied in PAMP in general and this project in particular. The National Graduate School in Scientific Computing is also sponsored by SSF. High performance is a key factor in scientific computing and SMPs are often being used to obtain high performance. Our performance prediction and visualization tool has been evaluated on a set of scientific programs [2], and we believe that some of the guidelines obtain in this project will be applicable to scientific programs. The Excellence Center in Computer Science and Systems Engineering (ECSEL) aims at reducing the development time and cost for new products. Reduced development cost and time is also a major concern within PAMP. However, PAMP focuses on the particular problems of using SMPs in real-time applications.

11. Context

11.1 The research group

The research group consists of: Lars Lundberg, Håkan Grahn, Magnus Broberg, and Daniel Häggander.

11.2 Complementary activities and funding

We have funding from KK-stiftelsen for a project looking at cluster systems. The Ph.D. student in that project (Charlie Svahnberg) is working together with Daniel regarding some aspects on cluster systems.

11.3 Research cooperation

Professor Per Stenström (pers@ce.chalmers.se) acted as Ph.D. supervisor and examiner for Magnus Broberg up until his Licentiate degree. Daniel Häggander was enrolled as a Ph.D. student at Uppsala with professor Hans Hansson as supervisor up until his Licentiate degree. Daniel and Magnus has also participated in the graduate student conferences and summer schools arranged by ARTES.

Besides the research cooperation with the PAMP/ARTES nodes, we will work with professor Jan Bosch and his group at Karlskrona/Ronneby. Jan Bosch leads a research group in software engineering with focus on object-oriented software architectures (see http://www.ide.hk-r.se/~bosch/ARCS.html). He will serve as a local expert on software development methods, particularly maintainability and reusability aspects.

11.4 Industrial cooperation

Our industrial partner is Ericsson Software Technology and we are currently working together with two business units. The contact persons are: Katti Sundelin, Ericsson Software Technology AB, Soft Center, S-372 25 Ronneby, Sweden, phone: 0457-77583, Katti.Sundelin@epk.ericsson.se (main contact). Kennet Kjellsson, Ericsson Software Technology AB, Box 518, S-371 23 Karlskrona, Sweden, phone: 0455-395248, kennet.kjellsson@epk.ericsson.se.

12. References

[1] E. D. Berger and R. D. Blumofe, "Hoard: A Fast, Scalable, and Memory-Efficient Allocator for Shared Memory Multiprocessors", http://www.hoard.org.

[2] Magnus Broberg, Lars Lundberg, and Håkan Grahn, "VPPB - Visualization and Performance Prediction Tool for Multithreaded Solaris Programs", in Proceedings of the 12th International Parallel Processing Symposium, March-April 1998.

[3] H. Chen, B. Shirazi, J. Yeh, H. Youn, and S. Thrane, "A Visualization Tool for Display and Interpretation of SISAL Programs", Proc. ISCA Int'l Conf. on Parallel and Distributed Computing Systems, Oct. 1994.

[4] W. Diestelkamp and L. Lundberg, “Performance Evaluation of a Generic Database Systems”, International Journal of Computers and Their Applications, September, 2000.

[5] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, “Design Patterns - Elements of Reusable Object-Oriented Software”, Addison-Wesley, 1995.

[6] W. Gloger, “Dynamic memory allocator implementations in Linux system libraries”, *http://www.dent.med.uni-muenchen.de/~wmglo/malloc-slides.html*.

[7] J. K. Hollingsworth and B. P. Miller, “Dynamic Control of Performance Monitoring on Large Scale Parallel Systems”, Proc. Int’l Conf. on Supercomputing, pp. 185-194, Jul. 1993.

[8] Daniel Häggander, and Lars Lundberg, “Optimizing Dynamic Memory Management in a Multithreaded Application Executing on a Multiprocessor”, in Proceedings of the International Conference on Parallel Processing, August 1998.

[9] D. Häggander, PO. Bengtsson, J. Bosch, and L. Lundberg, *Maintainability Myth Causes Performance Problems in SMP Applications*, in Proceedings of the 6th IEEE Asia-Pacific Software Engineering Conference, Takamatsu, Japan, December 1999.

[10] E. Kraemer and J. Stasko, “The Visualization of Parallel Systems: An Overview”, J. of Parallel and Distributed Computing, Vol. 18, pp. 105-117, 1993

[11] S. Lei and K. Zhang, “Performance Visualization of Message Passing Programs Using Relational Approach”, Proc. ISCA Int’l Conf. on Parallel and Distributed Computing Systems, pp. 740-745, Oct. 1994.

[12] V. Pillet, J. Laboarta, T. Cortes, and S. Girona, “PARAVER: A Tool to visualize and Analyse Parallel Code,” University of Politencia, Catalonia, CEPBA/UPC Report No. RR-95/03, Feb. 1995.

[13] D. Lea, “A memory allocator”, http://g.oswego.edu/dl/html/malloc.html.

[14] Microquill, “SmartHeap for SMP”, http://www.microquill.com/smp

[15] S. R. Sarukkai and D. Gannon, “SIEVE: A Performance Debugging Environment for Parallel Programs,” J. of Parallel and Distributed Computing, Vol. 18, pp. 147-168, 1993.

[16] Connie U. Smith, “Performance Engineering of Software Systems”, Addison-Wesley, 1990.

[17] Z. Segall and L. Rudolph, “PIE: A Programming and Instrumentation Environment for Parallel Processing,” *IEEE Software*, 2(6):22-37, Nov. 1985.’

[18] P. Wilson, M. Johnstone, M. Neely, and D. Boles, “Dynamic storage allocation: A survey and critical review”, in Proceedings of 1995 International Workshop on memory Management, Kinross, UK, 1995 (Springer)

[19] B. Zorn, “Malloc/Free and GC implementations”, http://www.cs.colorado.edu/~zorn/Malloc.html.

Activity Report for 1999 for the project: “Design Guidelines and Visualization Support for Developing Parallel Real-Time Applications” P4-9805

Lars Lundberg
Department of Software Engineering and Computer Science, University of Karlskrona/Ronneby,
Soft Center, S-372 25 Ronneby, Sweden, Phone: +46-(0)457-385833, Fax: +46-(0)457-271 25
Lars.Lundberg@ipd.hk-r.se

1. Background

Two Ph.D: students - viz. Daniel Häggander and Magnus Broberg - are currently working in this project. Both of them started somewhat before 1999. Magnus and Daniel took their Licentiate degrees during 1999 from Chalmers and Uppsala respectively. They are, however, now Ph.D. students at Karlskrona/Ronneby.

2. Comparison with plan and other aspects

The expected results, as stated in the project plan, were:

- a set of guidelines on how to design parallel real-time applications with high, predictable and scalable performance. The guidelines will also consider reusability and maintainability aspects.
- a tool for visualizing and predicting the performance of parallel programs. The tool will make it possible to predict the execution time of and identify performance bottlenecks in parallel applications.

The licentiate thesis by Daniel Häggander contained ten design guidelines for migrating server applications to SMPs. This is very close to the goal in the first bullet above. One finding by Daniel was that dynamic memory management can cause severe serialization bottlenecks when using a multiprocessor. Daniel’s work will now continue along two parallel paths: first to develop a method that automatically solves the dynamic memory problem in the compilation phase. Second, to validate his guidelines in an industrial project at Ericsson. He is currently working with these two activities. This is somewhat different from the (very rough) outline that we had in the project plan. In the original plan the next step should be to consider other execution environments than Sun Solaris, e.g. Windows NT. We feel that the current line of research is, however, more promising for several reasons, i.e. it would have a large industrial impact if we could solve the dynamic memory problem transparent to the programmer in the compilation phase, and we think that it is important to do a thorough validation of the guidelines in a real industrial environment.

The licentiate thesis by Magnus Broberg was based on a tool for visualizing and predicting the performance of parallel programs. The tool was developed by Magnus as a proof of concept for some of his ideas. Magnus is also following the project plan very closely after his Licentiate degree, i.e. he is currently considering execution environments other than Sun Solaris. The cross-simulation technique for Linux is the first example of this (see reference [M6] below).

3. List of publications

Publications by Magnus:

M1. M. Broberg, L. Lundberg and H. Grahn, *VPPP - A Visualization and Performance Prediction Tool for Multithreaded Solaris Programs*, in Proceedings of 12th International Parallel Processing Symposium, Orlando, April 1998.

M2. M. Broberg, L. Lundberg, and H. Grahn, *VPPB - The Second Version: An Approach to Monitor Solaris Kernel Threads*, in DSA-98 Seventh Swedish Workshop on Computer Systems Architecture, Göteborg, Sweden, June 3-5, 1998.

M3. M. Broberg, L. Lundberg and H. Grahn, *Visualization and Performance Prediction of Multithreaded Solaris Programs by Tracing Kernel Threads*, in Proceedings of 13th International Parallel Processing Symposium, San Juan, Puerto Rico, April 1999, pp. 407-413.

M4. M. Broberg, L. Lundberg, and H. Grahn, *Performance Optimization Using Extended Critical Path Analysis in Multithreaded Programs on Multiprocessors*, submitted for journal publication, in second review.

M5. M. Broberg, *An Approach for Performance Tuning of Multithreaded Applications on Multiprocessors*, Licentiate Thesis, Chalmers University of Technology, Sweden.

M6. M. Broberg, *Performance Tuning of Multithreaded Applications for Multiprocessors by Cross-Simulation*, in Proceedings of the ISCA International Conference on Parallel and Distributed Computer Systems, Las Vegas, August 2000, also presented at the ARTES student conference in Gothenburg, March 2000.

Publications by Daniel:

D1. D. Häggander and L. Lundberg, *Optimizing Dynamic Memory Management in a Multithreaded Application Executing on a Multiprocessor*, in Proceedings of International Conference on Parallel Processing, Minneapolis, US, August, 1998.

D2. D. Häggander and L. Lundberg, *Memory Allocation Prevented Telecommunication Application to be Parallelized for Better Database Utilization*, in Proceedings of the 6th Annual Australasian Conference on Parallel and Real-Time Systems, Melbourne, November 1999 (Springer-Verlag), pp. 258-271.

D3. L. Lundberg, J. Bosch, D. Häggander, and PO. Bengtsson, *Quality Attributes in Software Architecture Design*, in Proceedings of the 3rd Annual IASTED International Conference on Software Engineering and Applications, Scottsdale, Arizona, October 1999.

D4. D. Häggander, PO. Bengtsson, J. Bosch, and L. Lundberg, *Maintainability Myth Causes Performance Problems in Parallel Applications*, in Proceedings of the 3rd Annual IASTED International Conference on Software Engineering and Applications, Scottsdale, Arizona, October 1999.

D5. D. Häggander, PO. Bengtsson, J. Bosch, and L. Lundberg, *Maintainability Myth Causes Performance Problems in SMP Applications*, in Proceedings of the 6th IEEE Asia-Pacific Software Engineering Conference, Takamatsu, Japan, December 1999.

D6. D. Häggander and L. Lundberg, *Multiprocessor Performance Evaluation of a Telecommunication Fraud Detection Application*, in Proceedings of the ARTES Graduate Student Conference, Västerås, pp. 33-39, May 1999.

D7. D. Häggander, *Software Design When Migrating to Multiprocessors*, Licentiate Thesis, Department of Computer Systems, Uppsala University, Sweden.

D8. D. Häggander and L. Lundberg, *Attacking the Dynamic Memory Problem for SMPs*, in Proceedings of the ISCA International Conference on Parallel and Distributed Computer Systems, Las Vegas, August 2000.

D9. D. Häggander and L. Lundberg, *A Simple Process for Migrating Server Applications to SMPs*, submitted for journal publication.

4. Statement from Industry

Concerning the work done by Daniel Häggander and Magnus Broberg in the project "Design Guidelines and Visualization Support for Developing Parallel Real-Time Applications"

Magnus and Daniel have actively contributed in the development of two prototype versions of a product called DMO. The purpose of DMO is to detect abnormal situations in the telecommunication network, e.g. that a cell has a very high rate of lost calls. Such situations generally indicate that there is some problem with the cell, e.g. the corresponding base station may need to be repaired. It is important that these indications are done in real-time, since erroneous cells may need to be shut off from the rest of the network. The amount of information that needs to be monitored is enormous even for a medium sized network operator. It is therefor necessary to use computer based applications such as DMO for interpreting the data in real-time.

The DMO prototypes were developed by two groups of software engineering students at the University of Karlskrona/Ronneby. Ericsson handled the requirement specification and part of the project management. One of the groups developed a multithreaded version of DMO, which was intended to scale-up on a symmetric multiprocessor (a Sun SMP). Magnus Broberg's visualization tool played an important role in this development. The other group was working with a commercial DBMS. The design guidelines developed by Daniel Häggander were very valuable for both groups. The DMO project is an example of how Ericsson is cooperating with the University in research and education.

In order to collect information for his design guidelines, Daniel has participated in a number of development projects at Ericsson, e.g. the development of a real-time system for detecting fraud attempts in telecommunication networks. Daniel is currently participating in a project for developing a very performance demanding application for prepaid billing of cellular calls. The advice and expertise that Daniel has shared with us during these projects has been very valuable.

Karlskrona

Kennet Kjellsson
Business Center Manager
New Solutions

Ronneby

Katti Sundelin
Business Unit Manager,
Rating Engine

Short C.V. for Lars Lundberg

Name: Lars Lundberg, 620808-3359

URL: http://www.ide.hk-r.se/~lasse

Family: Married to Ulrika, a son Oscar born in 1997

Exams: M.Sc. (Civ. Ing.) Computer Science, Linköping University, 1986
Tekn. Lic. Computer Engineering, Lund University, 1990
Ph.D. (Tekn. Dr.) Computer Engineering, Lund University, 1993

Selected as a competent candidate for a professor position (kompetensförklarad) in Computer Engineering at Mälardalen University, Sweden (Mälardalens Högskola), 1997.

Selected as a competent candidate for a professor position (kompetensförklarad) in Computer Engineering at Halmstad University, Sweden (Högskolan i Halmstad), 1999.

Selected as a competent candidate for a professor position (kompetensförklarad) in Computer Sytems Engineering at University of Karlskrona/Ronneby, Sweden (Högskolan i Karlskrona/Ronneby), 1999.

Professional career:

Programmer of embedded systems, Elektronikcentrum i Svängsta during the summer and fall of 1986 (7.5 months).

Ph.D. student (doktorand), dept. of Computer Engineering, Lund University from January 1987 to May 1991. The position consisted of 75% research and 25% teaching and course development work.

Assistant professor (högskoleadjunkt) (50%) at University of Karlskrona/Ronneby (HKR) andconsultant (50%) at Elektronikcentrum i Svängsta from May 1990 to July 1991.

Assistant professor (högskoleadjunkt) at HKR. From July 1991 to May 1993.

Assistant professor (högskolelektor) in Computer Engineering at HKR from May 1993.

Head of department (prefekt) from July 1994 to January 1999. Deputy head of department (biträdande prefekt) from January 1999 to July 1999.

Associate professor (biträdande professor) in Computer Engineering at HKR from November 1997 to January 1999.

Professor in Computer Systems Engineering at HKR from January 1999.

Dean (dekanus, dvs. ordförande i fakultetsnämnden) of the technical faculty at HKR from April 1999.

Program comittee work:

- DSA-98 (Datorsystemarkitektur, a national swedish conference, Gothenburg, June 1998)
- AoM/IAoM-99 (17th Annual AoM/IAoM International Conference on Computer Science, sponsored by the Association of Management/International Association of Management, San Diego, August 1999)
- SEA'99 (3rd Annual IASTED International Conference on Software Engineering and Applications, Scottsdale, Arizona, October 1999)
- SNART'99 (A swedish conference on real-time systems)
- NOSA'99 (A nordic workshop on software architectures)
- SNPD'00 (International Conference on Software Engineering Applied to Networking & Parallel/Distributed Computing, Reims, france, May 2000).
- SEA'00 (4th Annual IASTED International Conference on Software Engineering and Applications, Las Vegas, Nevada, November 2000)

M.Sc. Students

Finished M.Sc. sudents:
- Mikael Roos, 1997,
- Magnus Broberg, 1997 (best master thesis award)
- Daniel Häggander, 1998 (best master thesis award)

Active M.Sc. students:
- Mattias Johansson and Henrik Hermansson
- Henrick and Johan
- Per Lidén
- Per Otterström

Ph. D. Students

I am currently advising five Ph.D. students. Two of these have taken their Licentiate degree during 1999 (a Swedish degree halfway between M.Sc. and Ph.D.).

- Daniel Häggander, Lic 1999
- Magnus Broberg, Lic 1999
- Wolfgang Diestelkamp
- Charlie Svahnberg
- Kamilla Klonowska

Lic opponent:

Tomas Lundqvist,"A Static Timing Analysis Method for Programs on High-Performance Processors", Chalmers, June 1999.

Ph. D. comittee work:

Ashley Saulsbury, "Attacking Latency Bottlenecks in Distributed Shared Memory Systems", KTH, December 1999.

Evaluator:

- Assitent Professor (Lektor) in Computer Engineering, University College of Kristiantad, 1994.
- International research collaboration project in real-time systems, STINT 1998
- Assitent Professor (Forskarassistent) in Computer Engineering, Chalmers University, 2000.
- Assitent Professor (Lektor) in Computer Engineering, Chalmers University, 2000.
- Assitent Professor (Lektor) in Computer Engineering, Mälardalen University, 2000.
- Professor in Computer Engineering, Royal Institute of Technology, 2000.

Overview of Scientific publications (the numbers refer to the list below)

15 Journal articles:	1,3,14,17,19 (also published as a chapter in a book),21,24,25,26,27, 33,38,40,51,52
33 Conference papers:	2 (invited paper),5,7,9,12,15,16,18,20,22,28,29,30,32,34,35, 36,37,39,40,41,42,44,46,47,48,49,53,54,55,56,57,58
3 Workshop papers:	8,11 (invited paper),43
5 Technical reports:	6,10,23,31,50
2 Theseses:	4,13
1 Monograph:	45 (in preparation)

Scientific publications and reports in chronological order (approximately)

1. A Parallel Ada System on an Experimental Multiprocessor

L. Lundberg, Software Practice and Experience, Vol 19(8), 1989, pp. 787-800. (included in references 4 and 13)

2. The MUMS Multiprocessor Ada Project

L. Lundberg and A. Ardö, in Proceedings of the Distributed Ada Symposium, Southampton, December 1989, pp. 243-266 (invited paper). (included in reference 4)

3. A Protocol to Reduce Global Communication in Distributed Ada Tasking

L. Lundberg, Journal of Parallel and Distributed Computing 1990. (included in reference 13)

4. Implementation of a Parallel Ada System and Evaluation of some Performance Improvement Techniques

L. Lundberg, Thesis for the degree of Teknisk Licentiat, March 1990, Department of Computer Engineering, Lund University, Sweden.

5. Static Process Allocation using Information about Program Behavior

L. Lundberg, in Proceedings of the 24’th Hawaii International Conference on System Sciences, Hawaii, January 1991, pp. 1-9. (included in reference 13)

6. Performance Efficient Reuse in Multiprocessors

L. Lundberg, REBOOT-report no. 3001, February 1991 (ESPRIT-project).

7. A Coprocessor for High Performance Multiprocessor Ada Tasking

L. Lundberg, in Proceedings of the Ada-Europe 1991 Conference, Athens, May 1991, Springer-Verlag, pp. 147-165. (included in reference 13)

8. Reuse of Software Components in Real-Time Systems

L. Lundberg, in Proceedings of the first National Swedish Symposium on Real-Time Systems, Uppsala, August 1991, pp. 74-80.

9. *A Lookup-free Multiprocessor Cache Design.*

P. Stenström, F. Dahlgren and L. Lundberg, in Proceedings of International Conference on Parallel Processing, Chicago, August 1991.

10. *Reuse Methodology Applied to Concurrency - Performance Issues*

L. Lundberg, REBOOT-report no. 3012, August 1991 (ESPRIT-project).

11. *A set of tools for designing high performance distributed Ada systems*

L. Lundberg, presented at a NATO Workshop on Object-Orientation and Distributed Systems, Quebec, May 1992 (invited paper).

12. *Predicting the Speedup of Parallel Ada Programs*

L. Lundberg, in Proceedings of the Ada-Europe 1992 Conference, Amsterdam, June 1992, Springer-Verlag, pp. 257-274. (included in reference 13)

13. *Parallel Program Scheduling – from Implementation Aspects to Performance Bounds*

L. Lundberg, Doctoral thesis, May 1993, Department of Computer Engineering, Lund University, Sweden.

14. *Performance Bounds on Multiprocessor Scheduling Strategies for Statically Allocated Programs*

L. Lundberg, the journal BIT, June, 1993, pp. 190-213. (included in reference 13)

15. *Performance Evaluation of Parallel Ada Programs using an Experimental Multiprocessor*

L. Lundberg, in Proceedings of the Ada-Europe 1993 Conference, Paris, June 1993, pp. 280-297.

16. *Neighbor Scheduling of Statically Allocated Parallel Programs*

L. Lundberg, in Proceedings of the Sixth International Conference on Parallel and Distributed Systems, October, 1993, Louisville, USA, pp. 92-96.

17 *Performance Bounds on Multiprocessor Scheduling Strategies for Chain Structured Programs*

L. Lundberg, Journal of Parallel and Distributed Computing 23, 1994, pp. 112-118.

18. *An Optimal Upper Bound on the Minimal Completion Time in Distributed Supercomputing*

L. Lundberg and H. Lennerstad, in Proceedings of the 1994 ACM International Conference on Supercomputing, July, 1994, Manchester, England, pp. 196-203.

19. *Optimal Scheduling Results for Parallel Computing*

H. Lennerstad and L. Lundberg, in SIAM News, August/September, 1994. Also updated and published as chapter 14 in the book “Applications on Advanced Architecture Computers” (ed. Greg Astfalk), 1996, ISBN 0-89871-368-4.

20. *An Optimal Lower Bound on the Maximum Speedup in Multiprocessors with Clusters*

L. Lundberg and H. Lennerstad, in Proceedings of the first International Conference on Algorithms and Architectures for Parallel Processing 1995, April, Brisbane, Australia, pp. 640-649.

21. *An Optimal Execution Time Estimate of Static versus Dynamic Allocation in Multiprocessor Systems*

H. Lennerstad and L. Lundberg, SIAM Journal of Computing, Vol. 24, No. 4, pp. 751-764, August 1995.

22. *An Optimal Bound on the Gain of using One Large Processor Cluster instead of Number of Small Clusters*

L. Lundberg and H. Lennerstad, in Proceedings of the ISCA International Conference on Parallel and Distributed Computing Systems, Orlando, Florida, September, 1995.

23. *An Optimal Bound on the Gain of Using Dynamic versus Static Allocation in Multiprocessor Computers*

L. Lundberg and H. Lennerstad, Technical report, April 1993, dept. of Computer Engineering, Lund University. (included in reference 13)

24. *Bounding the Gain of Changing the Number of Memory Modules in Shared Memory Multiprocessors*

L. Lundberg and H. Lennerstad, Nordic Journal of Computing, Vol. 4, No. 3, 1997, pp. 233-258.

25. *Optimal Combinatorial Functions Comparing Multiprocess Allocation Performance in Multiprocessor Systems*

H. Lennerstad and L. Lundberg, SIAM Journal of Computing, Vol. 27, No. 6, 2000, pp. 1816-1838.

26. *Optimal Worst Case Formulas Comparing Cache Memory Associativity*

H. Lennerstad and L. Lundberg, SIAM Journal of Computing, to appear.

27. *Combinatorial formulas for optimal cache memory efficiency*

H. Lennerstad and L. Lundberg, SIAM News, Volume 29, Number 6, July/August, 1996.

28. *Combinatorics for multiprocessor scheduling optimization and other contexts in computer architecture*

H. Lennerstad and L. Lundberg, in Proceedings of the Conference of Combinatorics and Computer Science, Brest, France, 1995, Lecture Notes in Computer Science, Springer-Verlag.

29. *Multiprocessor Performance Evaluation of Billing Gateway Systems for Telecommunication Applications*

L. Lundberg, in Proceedings of the ISCA International Conference on Parallel and Distributed Computing Systems, Dijon, France, September 25-27, 1996.

30. *Multiprocessor Performance Evaluation of Billing Gateway Systems for Telecommunication Applications*

L. Lundberg and D. Häggander, in Proceedings of the ISCA 9th International Conference on Computer Applications in Industry and Engineering, December, Orlando 1996.

31. BGw: A Parallel Real-Time System for Telecommunication Applications

L. Lundberg and D. Häggander, submitted for publication.

32. Comparing the Optimal Performance of Different MIMD Multiprocessor Architectures

L. Lundberg and H. Lennerstad, in Proceedings of 12th International Parallel Processing Symposium, Orlando, April 1998.

33. Using Recorded Values for Bounding the Minimum Completion Time in Multiprocessors

L. Lundberg and H. Lennerstad, IEEE Transactions on Parallel and Distributed Systems, pp. 346-358, April, 1998.

34. Predicting the Speed-up of Multithreaded Solaris Programs

L. Lundberg and M. Roos, in Proceedings of the IEEE Conference on High Performance Computing, Bangalore India, December 1997.

35. Bounding the Minimal Completion Time of Static Mappings of Multithreaded Solaris Programs

L. Lundberg, In proceedings of EuroPar-97, Passau, Germany, August 1997.

36. An Optimal Performance Bound on the Gain of Optimizing Data Layout in Vector Processors

L. Lundberg and D. Häggander, in Proceedings of the ACM International Conference on Supercomputing, Melbourne Australia, pp. 235-242, July 1998.

37. Evaluating the Performance Implications of Binding Threads to Processors

L. Lundberg, in Proceedings of the IEEE Conference on High Performance Computing, Bangalore India, December 1997.

38. Predicting and Bounding the Speedup of Multithreaded Solaris Programs

L. Lundberg, Journal of Parallel and Distributed Computing, June 1999, pp 322-333.

39. VPPP - A Visualization and Performance Prediction Tool for Multithreaded Solaris Programs

M. Broberg, L. Lundberg and H. Grahn, in Proceedings of 12th International Parallel Processing Symposium, Orlando, April 1998.

40. An Optimal Bound on the Gain of Permitting Dynamic Allocation of Communication Channels in Distributed Processing

L. Lundberg and H.Lennerstad, in Proceedings of the International Conference on Principles of Distributed Systems (OPODIS'97), Picardie, France, December 10-12, 1997. An extended version of this paper has been published in ACTA INFORMATICA (Springer), Vol 36, Issue 6, pp 425-446.

41. Fixed Priority Scheduling of Age Constraint Processes

L. Lundberg, In proceedings of EuroPar-98, Southampton, England, September 1998.

42. *Optimizing Dynamic Memory Management in a Multithreaded Application Executing on a Multiprocessor*

D. Häggander and L. Lundberg, in Proceedings of International Conference on Parallel Processing, Minneapolis, US, August, 1998.

43. *VPPB - The Second Version: An Approach to Monitor Solaris Kernel Threads*

M. Broberg, L. Lundberg, and H. Grahn, in DSA-98 Seventh Swedish Workshop on Computer Systems Architecture, Göteborg, Sweden, June 3-5, 1998.

44. *Multiprocessor Scheduling of Age Constraint Processes*

L. Lundberg, in Proceedings of the Fifth IEEE International Conference on Real-Time Computing Systems and Applications, Hiroshima, Japan, October, 1998.

45. *Optimal Scheduling Combinatorics*

H. Lennerstad and L. Lundberg, monograph for SIAM, the mongraph has not yet been formally accepted, but we are having a positive dialogue with the editor for this series of monographs (Peter L. Hammer). The monograph is under preparation.

46. *Visualization and Performance Prediction of Multithreaded Solaris Programs by Tracing Kernel Threads*

M. Broberg, L. Lundberg and H. Grahn, in Proceedings of 13th International Parallel Processing Symposium, San Juan, Puerto Rico, April 1999, pp. 407-413.

47. *Optimal Recovery Schemes for Fault Tolerant Distributed Computing*

L. Lundberg and C. Svahnberg, in Proceedings of the 6th Annual Australasian Conference on Parallel and Real-Time Systems, Melbourne, November 1999 (Springer-Verlag), pp. 153-167.

48. *Promis, a Generic Product Information Database System*

W. Diestelkamp and L. Lundberg, in Proceedings of the ISCA 14th International Conference on Computers and Their Applications, Cancun, Mexico, April 1999, pp. 53-58.

49. *Memory Allocation Prevented Telecommunication Application to be Parallelized for Better Database Utilization*

D. Häggander and L. Lundberg, in Proceedings of the 6th Annual Australasian Conference on Parallel and Real-Time Systems, Melbourne, November 1999 (Springer-Verlag), pp. 258-271.

50. *Performance Optimization Using Extended Critical Path Analysis in Multithreaded Programs on Multiprocessors*

M. Broberg, L. Lundberg, and H. Grahn, submitted.

51. Performance Evaluation of a Generic Database Systems

W. Diestelkamp and L. Lundberg, accepted for publication in the International Journal of Computers and Their Applications.

52. Utilization Based Schedulability Bounds for Age Constraint Process Sets in Real-Time Systems

L. Lundberg, Journal of Real-Time Systems, accepted for publication.

53. Quality Attributes in Software Architecture Design

L. Lundberg, J. Bosch, D. Häggander, and P-O. Bengtsson, in Proceedings of the 3rd Annual IASTED International Conference on Software Engineering and Applications, Scottsdale, Arizona, October 1999.

54. Maintainability Myth Causes Performance Problems in Parallel Applications

D. Häggander, PO. Bengtsson, J. Bosch, and L. Lundberg, in Proceedings of the 3rd Annual IASTED International Conference on Software Engineering and Applications, Scottsdale, Arizona, October 1999.

55. A Performance Dimensioning Guide for a Large Real-Time Telecommunication System

C. Svahnberg, L. Lundberg, and M. Roos, in Proceedings of the ISCA 15th International Conference on Computers and Their Applications, New Orleans, USA, March 2000.

56. Maintainability Myth Causes Performance Problems in SMP Application

Daniel Häggander, Per-Olof Bengtsson, Lars Lundberg, Jan Bosch, in Proceedings of APSEC ’99, the 6th IEEE Asian-Pacific Conf. on Software Engineering, Takamatsu, Japan, December 1999.

57. Switching Database Models in Order to Provide Tradeoffs Between Maintainability and Performance

W. Diestelkamp and L. Lundberg, in Proceedings of the ISCA 12th International Confernce on Computer Applications In Industry and Engineering, Atlanta, November 1999.

58. Attacking the Dynamic Memory Problem in SMPs

D. Häggander and L. Lundberg, in Proceedings of the ISCA International Conference on Parallel and Distributed Computing Systems, Las Vegas, USA, August 8-10, 2000, to appear.

Educational publications and reports (mostly in Swedish)

A. *Idéplan för ett nytt utbildningssystem vid HKR inom området elektro- och datateknik.*

Lars Lundberg, 1991.

B. *Grunderna i IT (kursbok, 350 sidor)*

L. Lundberg och M. Eriksson, Utbildningsradions förlag, 1996, ISBN 91-26-96394-9.

C. *Grunderna i IT distansstudiehandledning, Grundkurs IT 5 poäng*

C-J. Bachofner, L. Lundberg och H. O. Åkesson, Utbildningsradions förlag, 1996, ISBN 91-26-96396-5.

D. *Grunderna i IT CD-ROM så fungerar datorn och dess delar (The computer and its parts)*

Manus: L. Lundberg, U. Brennestig och S. Malmström, Utbildningsradions förlag, 1996, ISBN 91-26-96397-3.

E. *Objektorienterad programmering i Java - distansstudiehandledning*

Lars Lundberg, Utbildningsradions förlag, 1997, ISBN 91-26-97726-5

F. *Välkommen till Informationsteknikprogrammet 120 poäng - första året på distans*

Lars Lundberg, 1997.

G. *Diplom för Magnus Brobergs magisterabete som jag handledde.*
Förutom diplomet fick Magnus 10,000 kr.

H. *Kapitel 8 i boken Risker i Tekniska system*

Bo Arnesjö, Göran Collste, Bo Helgesson och Lars Lundberg, Utbildningsradions förlag, 1998, ISBN 91-26-98200-5.