Appendix D

Research Plan

1998-2002

for

PAMP

Symmetric Multiprocessors in High-Performance Real-Time Applications

April 19, 1998

Contact person:

Per Stenström, Department of Computer Engineering, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden. Phone: +46-31-772 1761, Fax: +46-31-772 3663, Email: pers@ce.chalmers.se, WWW http://www.ce.chalmers.se/pamp

SUMMARY OF RESEARCH PLAN

The objective of this collaborative research program is to provide design methods to use symmetric multiprocessors (general-purpose, high-performance computing platforms) as a key technology to meet performance demands of emerging high-performance, real-time applications. Important real-time aspects in this context are Quality-of-Service demands; a high and at the same time guaranteed performance level is a part of the system specification. A consortium of four industrial partners (Ericsson Software Technology, Ericsson UAB, Ericsson CSlab, and Prosolvia/Clarus) with application domain knowledge and four academic partners (Chalmers, Karlskrona/Ronneby, SICS, and Uppsala) with technology expertise has jointly developed this research plan. In particular, performance demands of transaction processing and multimedia applications are targeted in this research plan.

The expertise of the partners in this consortium provides a solid ground for advancing state-of-the art of symmetric multiprocessing technology. It also serves for efficient dissemination of results to end-users in industry with a strong need to use symmetric multiprocessors to accommodate the high-performance requirements of future products. The project focuses on software methods (performance prediction/tuning, parallelization and operating system concepts) as well as hardware methods (multiprocessor architecture tradeoffs and high-speed networking concepts). Thus, the consortium will address a wide range of aspects in the design of high-performance systems for a large number of industrially important applications. Collaboration between nodes not only takes the form of using the same methodology but also allows the project as a whole to take a multidisciplinary approach to address the issues in an industrial high-performance computer system.

The PAMP research program is a separate program within the SSF-funded ARTES program with separate funding. While the project costs are higher, the asked funding amounts 2, 4, 5, 5, 5 MSEK per year from 1998-2002, respectively. The program is intended to start on July 1, 1998.

Key words: Symmetric multiprocessors, transaction processing, multimedia, hardware and software system design, high-performance, quality of service.

1. Overall Ambition and Industrial Relevance

The objective of this collaborative research program is to provide methods to use symmetric multiprocessors as a key technology to meet the performance demands of emerging high-performance, real-time applications. A consortium of four industrial partners with application domain knowledge and four academic partners with multiprocessor technology expertise has jointly developed this research plan. This consortium also provides a solid ground for advancing state-of-the-art in symmetric multiprocessing technology and disseminating the developed methods to end-users in industry. The goal is that the program will enable new performance demanding applications with soft real-time, or Quality-of-Service, demands in the future. Application examples can be found today in transaction processing and multimedia, where a high throughput and a short response time must be guaranteed to maintain a high Quality-of-Service. It is this Quality-of-Service view of real-time requirements that is focussed on in this program. In addition, this view provides a complement to the predominant hard real-time view in other ARTES-related projects.

IT industry has enjoyed a dramatic increase in performance of mainstream microprocessor technology over the last few decades. Very recently, single processor systems have rapidly been replaced by computer systems using multiple microprocessors, called *symmetric multiprocessor systems* (SMP), to meet the tremendous performance requirements of emerging industrial applications. SMP technology offers a significantly higher performance for a wide range of applications because it allows an incremental increase of job throughput and a decreased response time by simply adding more microprocessors in a modular fashion. Therefore, it is not surprising that SMP technology is today an enabling computer technology for many important application domains in information processing (transaction processing and decision support systems), embedded systems (VME and SCI-based systems) and multimedia (media/web servers, virtual reality, and computer graphics). The importance of this technology is also expected to grow dramatically in the future with the emergence of microprocessors supporting thread-level parallelism.

From an application's point-of-view, SMP technology also offers an intuitive programming interface that makes it possible to reuse software originally developed for traditional single-processor systems. While it is fairly easy to port existing applications from a functional point of view, performance tuning with the goal of achieving a high and predictable performance level across platforms is however difficult. To aid designers of industrial high-performance, real-time applications in making their applications meet required performance levels, this collaborative research program aims at developing design methods that simplify the design of industrial applications using SMP platforms. The results from the project are expected to enable new application areas and bring down system development costs, thus increasing the competitiveness of emerging high-performance industrial applications. While the results are expected to be applicable to a wide range of application domains, the consortium will focus on transaction-oriented applications in the telecommunication and multimedia domains; important emerging high-performance application domains for the Swedish IT industry.

This program draws on the academic competence in symmetric multiprocessor technology, that has been developed over more than a decade in earlier research programs, and on application domain knowledge of the participating companies. This gathered experience will now be exploited to develop hardware and software design methods and applying them to industrial high-performance, real-time applications. Besides publishing results in journals and conferences of high international calibre, the results will be transferred to industry as documented methods, test implementations, and demonstrations.

In the framework of a 5-year research program, the participating project partners will focus on methods to achieve a high and predictable application performance on a range of SMP system organizations. The research is expected to produce results in the following areas (examples of concrete results are given in brackets):

- Design principles and performance characterizations of SMP systems <methodology for parallelization of sequential application to improve resource utilization, methodology for determining system design parameters (processor/memory organization) to meet performance and Quality-of-Service demands>;
- Performance characterizations of industrial high-performance, real-time applications and impact of system properties on performance <basic knowledge about program behavior of transaction-oriented and multimedia applications on SMP systems>
- Performance prediction methodologies and design methods <performance evaluation, scheduling, and performance tuning methods, hardware and software design methods for performance enhancements and Quality-of-Service maintenance>;
- Design principles and performance characterizations of parallelization and memory management policies <parallelization and scheduling algorithms for incorporation in commodity software systems, such as Posix-compliant operating systems>; and
- Network interface and system software design methods for I/O demanding SMP applications <run-time support and parallel protocol processing methods>.

The consortium consists of the following academic institutions: Chalmers, Karlskrona/Ronneby, SICS, and Uppsala, and the following companies: Ericsson Software Technology, Ericsson UAB, Ericsson CSlab, and Prosolvia/Clarus. The academic partners will develop methods and tools to support the design of applications to effectively utilize symmetric multiprocessor systems. The end-user companies will provide application domain knowledge so that methods can be applied and demonstrated on challenging real-time applications. The academic nodes will provide technologies (programming and system design methods, runtime, compiler, and virtual reality (VR) technology), and the companies will provide application, technological and commercial knowledge during the project, and are expected to develop the results after the project.

2. Objectives and Justification

The rapid technology improvement of computer systems has quickly been exploited by new demanding applications. Virtually all applications in industrial computerized systems have real-time demands because they either interact with humans or with other computerized systems. Such real-time demands not only take the form of a high capacity, a low system development cost is also important.

One example is computer platforms used as an integral part of the data- and telecommunication infrastructure. Typically, such computer platforms take care of incoming transactions, process them, and send them on to other computer platforms. To assess the performance level in this case necessitates the need to understand how a transaction is processed by the interacting software and hardware components of the computer platform. Another example of an application domain that is quickly emerging is support for real-time computer graphics that is needed to process realistic images in real-time in multimedia applications such as virtual reality (VR). Common for these two application examples is that they have inherent parallelism and thus have a potential to meet their performance demands by symmetric multiprocessing.

There is a strong economic argument to use mainstream computer technology instead of using tailor-made systems for challenging applications. Symmetric multiprocessors constitute an enabling technology particularly suitable for these performance demanding application domains. As we shall see, compared to other parallel computers, such systems simplify application development and offer a high and scalable performance level in a cost-effective way.

2.1 Symmetric Multiprocessors: Enabling Technology for High-Performance Applications

Until recently, single-microprocessor systems have formed the basis for general-purpose computing platforms. Today, however, *symmetric multiprocessors (SMPs)* dominate as compute servers and constitute the main platform for systems on the medium and high end. The importance of this emerging technology will increase in the future because future microprocessors will most likely use symmetric multiprocessing as the main paradigm to sustain their performance growth. SMPs combine multiple processors that share the same memory address space and can be considered as an integrated computing platform with a potential performance level proportional to the number of processors. Commercial SMP systems are currently being used in the following application domains:

- *Scientific and engineering applications:* Important SMP products are offered by e.g. HP/Convex, Silicon Graphics/Cray, and Sun Microsystems.
- *Database and transaction processing (OLTP and decision support systems).* SMP technology is offered by e.g. Compaq/DEC, HP/Convex, IBM, Sequent Systems, and Sun Microsystems.
- *Embedded applications.* These are built around widely used backplane-bus standards such as FutureBus, SCI, and VME, and thus offer a competitive cost level.

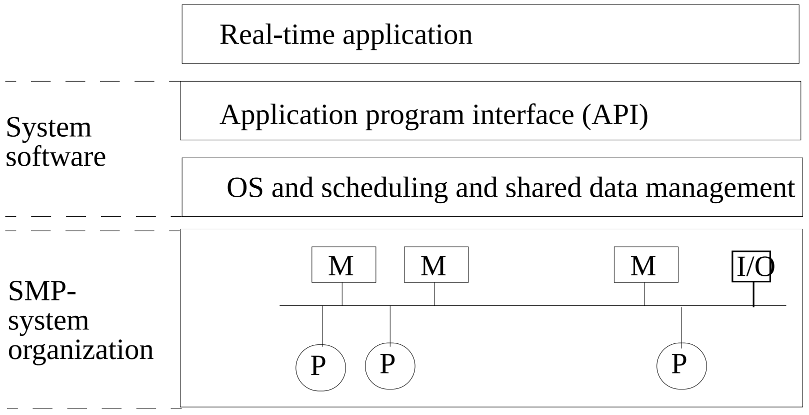
From the point of view of performance of e.g. transaction processing systems, SMPs offer two advantages. First, each individual transaction can be processed at a shorter time because the inherent parallelism in the processing of each transaction can be exploited. Second, to increase the transaction throughput, independent transactions can be processed by different microprocessors. Thus, to reach a higher performance level, one option at the hardware level is to increase the number of processors. In a VME-based system, for example, one can add more processor boards to the back-plane bus. Thus, SMP technology has a potential to exhibit a scalable performance growth as the performance requirements increase.

An ultimate goal of this research program is to provide technologies to use symmetric multiprocessors for applications that demand a high throughput and/or a short response time. Moreover, in terms of real-time requirements, the throughput and/or response time requirements must often be guaranteed to maintain a high Quality-of-Service. Systems with such real-time requirements are steadily increasing in the telecom and multimedia application domains.

From an application designer’s perspective, there is of course a strong incentive to reuse existing software components. Such software not only includes the application itself but also system software taking the form of commodity operating systems (OS). Owing to the single address space that SMPs provide, application as well as system software developed for single-processor systems can be ported to SMPs with limited efforts. While the difficulties from the programmer’s point of view lie in the partitioning of the computation into parallel processes, communication among cooperating processes is supported naturally through the shared memory semantics provided by SMP systems.

2.2 Research Challenges

While performance, cost, and technology trends speak in favor of SMP technology, designers using symmetric multiprocessing lack methods and design guidelines that can aid them to design applications and choose system organizations that guarantee the required performance level in an efficient way. The prevailing design methodology for high-performance systems is to take a holistic approach and consider the system as interacting layers of software and hardware components. This design methodology is adopted in this program. Consequently, the developed design methods and design principles will take into account issues involved in the application design, system software design, and the underlying system organization including the I/O system as illustrated in the diagram below.


Real-time application
Application program interface (API)
OS and scheduling and shared data management
System software
SMP-system organization
M
M
M
I/O
P
P
P

Methods are needed to help system designers to (1) design applications; (2) design the system software; (3) predict the performance and (4) to design the I/O subsystem with associated system software.

In designing a system that should meet the performance demands of the application, the following key issues arise:

- *Application performance issues.* Impact of the SMP system architecture on the parallelization strategy.
- *Scheduling and assignment issues.* Impact of the SMP system architecture and the quality-of-service requirements on scheduling and assignment strategies for the processes/threads in the application software.

- *Memory system performance issues.* Impact of the SMP memory system architecture on the application performance and design.
- *I/O system issues.* Impact of the I/O interfaces and protocol system software on the overall application performance.

Let’s review these issues in some more detail. In order to port a sequential application program to a multiprocessor, the application designer can either explicitly state what actions are to be carried out in parallel -- *explicit parallelism* -- or a parallelizing compiler could extract the *inherent parallelism* in the application, thus off-loading this burden from the designer. There are open issues associated with both options. For both approaches, the computation must be parallelized in a way that depends on the cost of coordination between parallel computations. This cost depends on the underlying system properties including OS implementations of coordination primitives as well as how they make use of the underlying system architecture, especially the memory system. To make good design trade-offs calls for the development of efficient parallelization methods that take into account the properties of the entire system including several layers of interacting hardware and software.

As for scheduling and assignment to achieve a high and predictable performance level, understanding the criteria to drive scheduling policies become important in the design of scheduling algorithms. Several options are possible for such scheduling decisions such as whether to consider static or dynamic placement of parallel computations. Moreover, in order to distribute the parallel computations in a way that maximizes system throughput, a scheduler of e.g. a transaction-oriented application must treat intra-transaction and inter-transaction parallelism differently. Since the ratio of access times between local and remote memories can be very high, the scheduling decisions depend on the placement of the data structures. Clearly, in order to make effective use of SMP technology, the design of OS primitives to express and coordinate computations must be addressed.

The third issue is concerned with coordination and communication among parallel computations. The shared-memory model provided by SMPs simplifies this task in that e.g. a shared data structure is accessible directly by all processes/processors. However, the partitioning of the data structure across one or several memory modules in the physical machine can have a dramatic impact on the performance. Typically, a memory hierarchy in an SMP has several levels: the on-chip cache, the off-chip cache, the local per-node memory and the remote memory. The ratio of access times for these levels can be 1:10:100:1000. Clearly, in order to design an application that can take advantage of the performance potential of SMPs calls for the development of methods that can aid the application designer to take the speed deviation of local and remote memories into account. Again, because coordination and, in some cases, communication is supported by OS level primitives, such methods must take into account the properties of the entire system including several layers of interacting hardware and software.

A network interface to a multiprocessor is different from a uniprocessor in some fundamental ways. The first issue is where it should be located. It may be shared by all processors or attached to one of them or there may be multiple interfaces attached to several processors that then need to be synchronized. The second issue is how communication data is distributed to the processors from and to the network interface(s). A multiprocessor has an internal interconnect that could be a performance bottleneck in the distribution of data to the appropriate processor. The interface will contend for the capacity of the interconnect with the other processors. A third issue is how protocol processing of higher layers can be distributed and parallelized over the processors.

A wide range of SMP systems are available. In its simplest form a number of microprocessors are connected to a number of memory modules by a shared bus as shown in the diagram. Such systems are available as industrial modular systems using VME and FutureBus as well as commodities from major high-end computer manufacturers. Multiprocessors using a distributed organization that can accommodate some hundreds of processors and targeting mainly the scientific domain also exist. Moreover, compute nodes connected together with LANs can also support symmetric multiprocessing by software layers on top of commodity platforms. Consequently, a wide range of platforms exist that can support the symmetric multiprocessing paradigm. All these implementations of the shared-memory model exhibit widely different timing models and pose a severe problem for software designers to achieve portability with respect to performance and Quality-of-Service requirements. The program should also focus on advancing state-of-the-art in providing methods to simplify performance tuning across platforms.

Finally, in order to address performance and real-time issues and evaluate the merit of a certain improvement technique, one must be able to perform application case studies and see how well e.g. Quality-of-Service requirements (possibly transformed into computation deadlines) are met. Unfortunately, in terms of identifying performance bottlenecks the prevailing practice is system measurements. Such measurements are often difficult to interpret because of limited observability of interactions between hardware and software components. In terms of addressing real-time requirements, analytical modeling is a prevailing methodology that often simplifies the timing model of the underlying hardware platform.

A promising methodology for performance as well as Quality-of-Service assessments is to use detailed simulation models of the target system; the application is executed on top of a clock-cycle true simulation model of the system. While this methodology promotes complete observability as to where the bottlenecks are located, a simplified view of the system environment has been assumed for the studied application domain. These models have not taken into account how the application interacts with the system software or with the incoming transactions. Moreover, while previous work has mainly focused on analyzing average performance for a system, prediction of bounds on e.g. throughput and response time are important in the kind of real-time applications that constitute the foundation for this program.

PAMP should therefore focus on further developing state-of-the-art performance prediction methodologies so that they can be applied to realistic system environments in which demanding real-time applications are considered. Apart from simulation-based techniques, such methodologies may also encompass analytical approaches and hybrids between them in addition to system measurements. The availability of quite different applications provided by industry will permit the collaborative project to focus on a wide range of aspects of SMP application analysis in real-time environments. Performance prediction tools and methods are also important when systems not available today are to be evaluated such as those that will be available at the marketplace in say 5-10 years from now.

To summarize, PAMP shall address issues related to how applications can achieve a high performance on a wide range of symmetric multiprocessor platforms. Aspects regarding how parallelization methods, scheduling and memory management policies interact with the system including I/O are therefore important. In addition, the project should emphasize and develop powerful methods to allow performance of improvement techniques to be predicted.

The results from PAMP are expected to take the form of methods and tools that can guide the development of application and system software for commodity systems that are available and expected to be of growing importance in the future. The research is driven by application domains that are specified by the industrial partners participating in the consortium. While the industrial partners will contribute with application domain knowledge and system requirement criteria, the academic partners will develop models of various system configurations and study the above mentioned design and performance issues in this framework. The collaboration between academic partners will enable a multidisciplinary approach in which a wide range of aspects of a high-performance real-time application can be addressed. As will be discussed in Section 3, the collaborating nodes also make use of the same methodology to reach a higher ambition.

3. Organization and Procedures

3.1 Participants

Four academic partners and four industrial participants have been involved in the planning phase of PAMP and are willing to participate in the project with the projects in Section 5. The academic partners are Chalmers, SICS, University of Karlskrona/Ronneby, and Uppsala and the industrial partners are Ericsson Software Technology (Soft Center, Ronneby), Ericsson UAB (Stockholm), and Ericsson CSlab (Stockholm), and Prosolvia/Clarus (Göteborg).

- The group at Chalmers, led by Per Stenström, has carried out research activities in SMP technology for more than a decade. Prosolvia/Clarus the Chalmers group will focus on parallelization and scheduling methods to meet performance and Quality-of-Service requirements of multimedia applications.
- Prosolvia/Clarus (Göteborg) develops Virtual Reality technology to be used in various important visualization applications such as virtual prototyping. In order to generate complex images at video rate, this technology needs very high performance. Together with Chalmers, they will look at how the performance requirements of computer graphics applications can be satisfied using SMP technology. (Contact person: Tomas Möller)
- The CNA-lab at SICS (Bengt Werner) has a strong background in modeling and analysing computer systems, especially shared memory architectures. They will develop modular simulation technology based on its simulator platform SimICS to enable accurate performance analysis of a parallel database system developed at Ericsson UAB.

- Ericsson UAB is studying implementations of network databases for future telecom products where SMP-technology plays an important role. Their interest in this project is to get access to the performance prediction methods developed at SICS and use it to pin-point bottlenecks across the hardware/software boundary in SMP platforms. (Contact person: Mikael Ronström)
- The group at University of Karlskrona/Ronneby, led by Håkan Grahn, has mainly done research in scheduling techniques and shared-memory system design. The aim of their work is to develop performance tuning methodologies based on visualization techniques that will aid application designers to develop more efficient parallel code based on threads models.
- Ericsson Software Technology has a long experience in designing large applications in the telecommunication domain. However, they have only recently started to look at parallel applications and how to use SMPs. Together with Karlskrona/Ronneby they will apply the performance tuning methods to a billing gateway application. (Contact person: Daniel Häggander)
- The Uppsala group led by Mats Björkman and Per Gunningberg will together with Ericsson CSlab study performance aspects of implementations of run-time support for network interfaces in SMP and parallel protocol implementations.
- Ericsson CSlab has developed the Erlang language for efficient implementation of applications with real-time demands. Together with Uppsala, they will study abstractions and mechanisms for host and network resource management, especially in the Exokernel environment, as well as parallel protocol implementations using Erlang. (Contact person: Håkan Millroth)

3.2 Approach

The approach taken to develop methods to exploit the performance potential of SMP technology is to use applications provided by the industrial partners as study objects or cases. Moreover, because the academic nodes cover a wide range of aspects of system design, PAMP makes it possible to reach a much higher ambition in terms of producing industrially useful results. Moreover, as a base for the collaboration many of the participating nodes will use the same design evaluation methodological approach. This approach is based on designing detailed timing models of a complete system with interacting hardware and software components. This will make it possible to transfer methods and tools among the participants and apply them to a wider range of application case studies.

Each project typically goes through a number of phases as follows. Functional and performance requirements of the applications will serve as sources for identifying relevant research issues to focus on. Based on the functional and performance requirements of the applications, the second phase of the project is devoted to development of concepts (design methods) aiming at shortening the design cycle for designers. The third phase, called the evaluation phase, aims at applying the methods to the applications provided by the industrial partners to understand the strengths/weaknesses of the developed methods. Finally, the fourth phase aims at refining and generalizing the concepts developed so as to make them applicable to a wider scope of applications/systems. In summary, each project is planned with identifiable milestones preferably according to the following template:

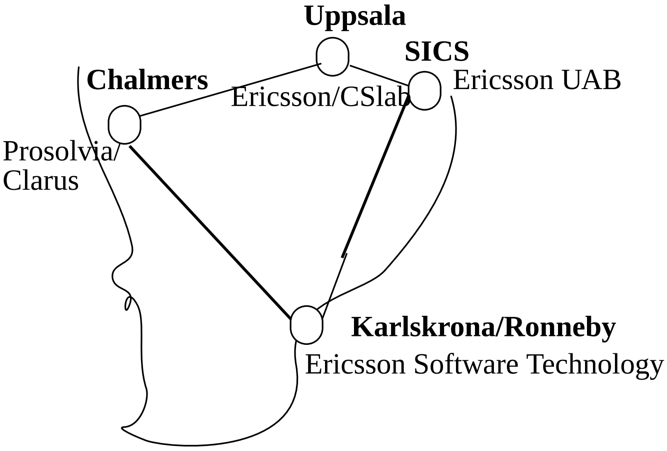
- Application analysis phase
- Conceptual development phase

- Evaluation phase
- Refinement and generalization phase

The first phase typically lasts less than a year whereas the bulk of the project is allocated to the following phases. Each Ph.D. student project is expected to last for 2+2 years, assuming 100% activity level and follows the general guidelines for ARTES projects with the supplementary guidelines in Section 5 of this document.

3.3 Program Organization and Management

The academic and industrial participants will cooperate in terms of intra- and inter-node activities as exemplified in the diagram below



Each node consists of one academic and industrial partner. The academic partner is responsible for the interactions and coordinations of the work done within each node. Each node then interacts with other nodes and these inter-node interactions are concerned mainly with transfer of methods and experiences among academic and industrial partners with the goal of broadening the impact of the experiences and results.

Within each node, the node coordinator (the academic partner) is responsible for establishing a work plan with identifiable work packages. These packages should preferably coincide with the milestones associated with the project phases in Section 3.2 and should be detailed in the formal project proposals according to the guidelines in Section 5.

PAMP will be administrated as a separate research program within the SSF-funded ARTES program where it acts organizationally as an ARTES project. To administrate PAMP, and to monitor the progress of the projects within PAMP, a steering committee will be established. This committee will consist of the program director of ARTES (Hans Hansson, Mälardalen University), one member from the ARTES board (Bertil Emmertz ABB/ISY), an industrial coordinator (Håkan Millroth Ericsson), and an academic coordinator (Per Stenström, Chalmers).

The overall task of the steering committee is to be responsible for the successful execution of the PAMP program. Among the subtasks to fulfill this goal, the steering committee shall review the initial plan of each project and carry out the review of the progress of each project against the milestones that have been defined according to the criteria in Sections 3.2 and 5. Moreover, it shall help in disseminating the results from all PAMP projects and give scientific advice to the project participants.

4. Results

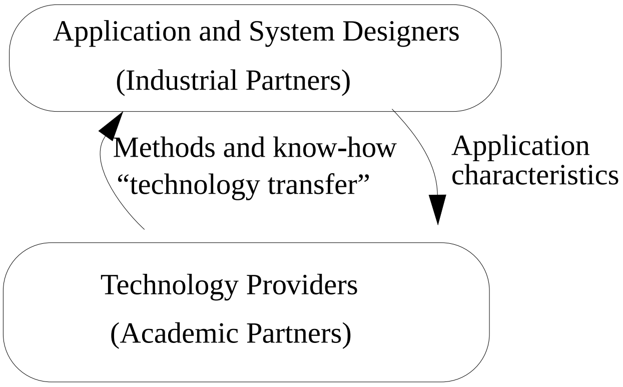
4.1 Industry Interaction and Technology Transfer

The activities in PAMP and the expected outcomes can be best understood from the diagram below that shows the interaction between the industrial and academic partners.

The industrial partners are involved in application development and system design. Most of the applications are transaction-oriented and adequate measures of performance are e.g. transaction throughput and response time. Unfortunately, in order to design a system for a given application that meets a certain performance goal, the designers lack adequate tools. Examples of such tools are performance prediction of the target system for architectural analysis and tools that aid the application designer to structure the software taking performance aspects of scheduling and shared data management into account. PAMP aims at providing industry with methods in addition to “know-how” in terms of guidelines for application and system design.

Academic outcomes:

- Software and hardware design methods to achieve a high application performance on SMPs
- Performance prediction methods for SMPs



The techniques and methods developed in the project will be useful for future applications with high performance demands. For such applications, symmetric multiprocessors are a key technology. Therefore, the know-how and methods disseminated by the project will act as an enabler for new applications. Examples of such applications are network-based information processing systems and multimedia. Those that have competence in how to accommodate the very high performance needed by new network-based services will have a strong advantage from a competition point of view.

4.2 Focus of the Program and Expected Industrial/Academic Results

The research will address issues detailed in Section 2.2. and the results are expected to be as follows:

- ***Design principles and performance characterizations of SMP systems <methodology for parallelization of sequential applications to improve resource utilization, methodology for determining system design parameters (processor/memory organization) to meet performance and Quality-of-Service demands>*** In order to reach a high performance speedup on SMP systems, the timing model of the underlying platform must be well understood. The goal here is to develop methods that can simplify parallel-program performance tuning across a wide range of SMP platforms.

- ***Performance characterizations of industrial high-performance, real-time applications and impact of system properties on performance <basic knowledge about program behavior of transaction-oriented and multimedia applications on SMP systems>.*** The multiprocessor research community has mainly focused on performance issues for scientific applications. At the same time, it is widely agreed that the behavior of them are fundamentally different from transaction-oriented and multimedia applications. PAMP will yield interesting results in identifying performance and real-time issues (Quality-of-Service requirements) for these application domains.
- ***Performance prediction methodologies and design methods <performance evaluation, scheduling, and performance tuning methods, hardware and software design methods for performance enhancements and Quality-of-Service maintenance>.*** One unique aspect of PAMP is to use architectural simulation techniques as the underlying methodology to assess performance and Quality-of-Service requirements. This approach will be further developed and applied to the applications studied. Another set of embryonic ideas that will be pursued in PAMP are to use real-time scheduling approaches and performance tuning techniques based on visualization to industrial applications with Quality-of-Service requirements.
- ***Design principles and performance characterizations of parallelization and memory management policies <parallelization and scheduling algorithms for incorporation in commodity software systems, such as Posix-compliant operating systems>.*** Some of the projects within PAMP aim at integrating scheduling algorithms into commodity Unix-operating systems. Examples of such projects are integration of an adaptive real-time scheduler, to be developed at Chalmers, that aims at meeting Quality-of-Service requirements.
- ***Network interface and system software design methods for I/O demanding SMP applications <run-time support and parallel protocol processing methods>.*** Because of the growing importance of network-based services, the communication capability of an information processing server is important. By also focusing on how I/O systems in SMPs are to be designed and how protocol processing can be made more efficiently, the program will make new network-based services possible.

5. Planning

As mentioned in Section 3.2, each PAMP-project will be typically divided into four distinct phases: (1) application analysis (2) concept development (3) evaluation, and (4) refinement/generalization. Each phase lasts approximately one year (with the exception of the last phase, which lasts two years). During the first phase, the academic and industrial partners within each node will specify functional and performance requirements of the application used as a case. In the second phase, the academic partners will develop the concepts in terms of methods to aid system designers. Such methods are exemplified by performance prediction, scheduling, and parallelization strategies. The third phase aims at applying the developed methods in application case studies to evaluate their merits. Finally, the fourth phase aims at further developing the methods with the goal of finding a wide applicability across application domains. Of course, this tentative project template can be adjusted to the specific needs in the project. However, to simplify coordination and progress review of the project, a template along these lines should be established for each PAMP project.

5.1 Project Selection Criteria

Two call-for-project-proposals will be announced (for projects intended to start in 1998 and 1999). Project funding is approved after a node has sent in a formal application according to the rules and criteria of ARTES projects (see ARTES program plan) and according to the following PAMP-specific criteria.

- A project shall target central research issues in the program plan. Examples are listed in Section 2.2. The proposal shall point out how it contributes to the goal of PAMP.
- A project shall involve at least two nodes of which one is industrial and one is academic. The research issues studied should be motivated from the application requirements provided by the industrial node.

5.2 Progress Review

Given the defined (expected milestones) of the project, the project is reviewed typically once a year. The review is based on a status report that is submitted to the steering committee of PAMP. A review hearing also takes place at the annual summer school organized by ARTES.

Each year a workshop is arranged within PAMP in which all the participants of the project are expected to attend. The workshop will summarize the experiences gathered so far and acts as an important means for cross-node information dissemination, especially for the industrial participants.

6. Projects

This section summarizes some proposed pilot projects that have been planned by the proposers of this program. Please note that they only serve as example projects in this document.

6.1 Chalmers

6.1.1 Objectives and Justification

The steady performance growth of mainstream computer technology has enabled new application areas such as multimedia. One example is virtual reality technology (VR) in which human actors interact with a simulated environment using different media such as images and sound. VR-technology is maturing and has made significant impact on CAD applications such as virtual prototypying and computer-aided training in surgery.

Because VR often involves simulation of physical systems and visualization in real-time, it is a performance-demanding technology. In fact, current high-performance microprocessors will not meet the enormous performance requirements needed to achieve a high realism in many emerging multimedia applications. Fortunately, simulation as well as visualization afford a lot of parallelism which makes symmetric multiprocessor systems interesting to consider.

The overall goal of the shared-memory multiprocessor project at Chalmers is to provide methods to meet the high-performance and quality of service (QoS) demands of multimedia applications using multiprocessor technology. We do this in two subprojects: parallelization strategies of computer graphics applications and distributed scheduling techniques. Together with Prosolvia/Clarus the focus of the work at Chalmers will be to provide design methods to reach a high and predictable performance for multimedia applications using a wide range of SMP platforms. Even though the project targets computer graphics applications, we expect the results to be applicable to a wide range of applications with high performance demands with real-time constraints.

This project is a continuation of Per Stenström's (the P.I. of this project) more than 10 years of research in SMP technology. His contributions to SMP technology are well recognized internationally in more than 50 scientific papers. He has a broad international contact net in the computer architecture community. He acts as editor for Journal of Parallel and Distributed Computing and acted as guest editor of a special issue of IEEE Computer on Shared-Memory Multiprocessing and a special issue of Proceedings of the IEEE on Distributed Shared Memory Systems. He has participated in SMP architecture research projects at Stanford University and at University of Southern California in the U.S. The project involves two faculty members, two Ph. D. students from Chalmers, and an industrial Ph. D student financed by Prosolvia/Clarus.

6.1.2 Approach

Two subprojects will be conducted, as follows:

1. Parallelization of computer graphics applications.

Computer graphics applications are an integral part of VR-technology. An important goal is that they must exhibit a high realism. Images must be computed under strict timing constraints at video rate. Our approach to exhibit a higher realism is to convert existing sequential computer algorithms to exploit the performance of SMP systems. Prosolvia/Clarus will provide interesting study objects and we will deliver programming methods to parallelize them in a platform-independent way as follows.

The main advantage of SMP systems is that it is fairly straight-forward to parallelize an application developed for single-processor systems to run correctly on a multiprocessor system. However, in order to reach performance levels close to what the underlying machine can deliver, substantial algorithmic changes are needed that take into account the performance model of the underlying architecture. In doing this, the programmer has to have great insights into the timing behavior of the architecture. Even worse, performance tuning is highly platform dependent and effectively prevents a tuned parallel program to be ported to another platform without significant performance losses. The goal of this subproject is to devise programming methods that make it possible to map applications developed for single-processor systems to a wide range of SMP platforms with limited effort so that the applications can take advantage of the performance potential of these platforms and yet limiting the performance tuning effort significantly. We want to pursue the following embryonic ideas in particular:

- Development of performance models and programming methods for a wide range of SMP platforms for computer graphics applications (The first 3 years)
- Generalizations/refinement of the programming methods (The last 2 years)

The goal of the first phase of the project is to develop a methodology for parallelizing computer graphics applications that take the timing models of a wide range of SMP platforms into account. The key observation that we want to explore is that there is a convergence in the performance models that future SMP platforms will rely on. This performance model is based on the notion of memory locality and exploits the fact that current and future SMP systems heavily rely on local memory hierarchies to perform well. The timing behavior of these memory hierarchies will be transferred in the project to programming methods that significantly reduce the performance tuning effort needed.

As a demonstration, we will run the same suite of parallelized computer graphics applications on a wide range of platforms and demonstrate the speedup in performance compared to parallelizations that do not take our methods into account. The goal of the second phase (year 4 and 5) is to generalize our findings by applying the programming methods to other application domains.

This project will be carried out by one Ph. D. student during five years.The methodology will be as follows. We will use a platform independent parallelization paradigm such as the ANL macro package to parallelize the computer graphics codes. We will identify performance bottlenecks of a straightforward parallelization using simulation platforms developed jointly with SICS. In addition, a final performance assessment will be conducted on real SMP systems available in the lab, at Prosolvia/Clarus as well as at the high-performance computer center at Chalmers (a 12-processor Sun Enterprise 4000 system, a 64-processor Sun Enterprise 10000, and a Silicon Graphics Origin 2000 system with 64 processors, and a DEC cluster interconnected by DEC memory channel).

2. Scheduling algorithms to meet Quality-of-Service requirements

A typical VR application ported to a multiprocessor system will typically consist of a number of parallel programs where there exist strict constraints on the time available to carry out each execution. Since the load on the system and the resources of the system vary over time, a key component to achieve high quality-of-service is a dynamic distributed real-time scheduling algorithm that can negotiate with the application about quality of service levels. The goal of this scheduling algorithm is to allocate resources in such a way that as high quality of service is achieved. While distributed static real-time scheduling algorithms have been developed in the past, the main problem is that they do not have the capability to trade QoS for computing resources. The goal of this work is to develop scheduling algorithms that make this happen.

Very little work on dynamic distributed real-time scheduling algorithms that meet QoS goals has been done. The goal of this project is to fill this gap. Our approach is to develop a methodology in which the QoS of an application is expressed as a function of the time available. Using this as input to a distributed real-time scheduler, the idea is to develop a strategy to select a computation and QoS levels depending on how much time and resources are available at each moment. This will drastically enhance the schedulability as well as the resource utilization of the system. As a result, the performance of the system can be better utilized. We envision the following milestones of this project:

- Methods to specify execution times for different Quality-of-service levels of computer graphics applications taking timing models of multiprocessor platforms into account (year 1)
- Dynamic distributed scheduling algorithms for real-time negotiations and demonstration of its performance based on computer graphics applications (year 2 and 3)
- Generalization/refinement of the scheduling methods (year 4 and 5)

Both projects will be carried out through collaborations with Prosolvia/Clarus. The competence at Chalmers in multiprocessor and real-time scheduling together with the computer graphics algorithm domain competence at Prosolvia/Clarus will be utilized to achieve the goal of the project.

6.1.3 Results

The expected scientific output of this project is

- Software design methods to map computer graphics applications for efficient execution on SMP systems in a platform independent way
- Parallel computer graphics algorithms.
- Scheduling algorithms for Quality-of-Service negotiations
- A framework for development and execution of parallel computer graphics applications to meet QoS requirements.

The technology transfer to Prosolvia/Clarus will be methods to use multiprocessor technology to meet the performance and QoS requirements of virtual-reality applications.

6.2 SICS

6.2.1 Objectives and Justification

Efficient databases is a key technology in controlling a telecom system. Ericsson UAB is currently implementing a parallel database on a shared memory multiprocessor composed of a cluster of multiprocessor workstations connected by SCI. Predicting and understanding the performance of such a system involves getting access to detailed statistics from the system e.g. cache hit ratios and resource contention. This knowledge is crucial to focus performance optimization on the true bottlenecks of the system.

Together with Ericsson UAB, SICS will extend an efficient simulator platform to enable measurement of this application. The resulting platform will be modular so that it can easily be adapted to evaluate other hardware architectures. The objective is thus to provide a modular, efficient and accurate performance measurement environment for multiprocessor systems. The technology transfer to Ericsson is knowledge how to measure performance accurately and a tool that supports it. The following subprojects have been identified as SICS responsibility:

- Adjusting the simulator to the Sparc V9 architecture. This will include UltraSparc specific features e.g. block load/store (referred to as sparc V9+).
- Creating a timing simulator for UltraSparc.
- Generalizing the timing simulator to support a multiprocessor.
- Generalizing the timing model to calculate best case and worst case timing.
- Building a memory hierarchy for the Ericsson prototype configuration including an SCI interface.
- Verifying the function and timing of the model with the prototype
- Simulator support for Solaris lightweight threads

The subprojects all aim at supplying a measurement environment. The project leverages on earlier experiences of performance measurements in cooperation with Ellemtel and processor simulator development of SparcV8 and Motorola 88000 architectures. Ericsson UAB is responsible for a number of subtasks within the project. The following have been identified:

- Database design
- UltraSparc Plex compiler
- Plex interpreter
- Performance measurement and optimization
- Adjusting the application software to run on the simulator
- Creating software/traces that generate load for the application

6.2.2 Application structure

The prototype is based on an existing database product in AXE 10. To maintain software compatibility an emulator for APZ, Ericsson’s proprietary hardware architecture, runs on a open workstation, currently Sparc. The shared memory is used to implement efficient message passing between APZ emulators. To speed up execution some software (which is written in Plex) can be complied directly to native Sparc code instead of being interpreted.

The system is typically 5-15 nodes but should allow for a large number (100-1000) of nodes to work concurrently. The prototype node is a 2 processor UltraSparc Server. The network is SCI which gives the system a distributed shared memory. Currently SCI is connected to the IObus but in the future It may be connected directly to the memory bus (for increased performance). The database may be accessed via an ATM interface or via SCI.

6.2.3 Approach

The simulator has a well defined application that should run on the simulator. This defines the minimal functionality that will be supported. The simulator supports emulation of parts of the Solaris operating system. This will thus not be simulated. While the user mode instruction set will be fully supported, including some extensions, not every possible system call will be supported.

Verification of the of the instruction set functionality will be automated as much as possible. In addition to this verification of the timing model can be done by comparing executions on the simulator with executions on a prototype system. The application can also run on the prototype system where some data can extracted e.g. execution time and hardware counter measurements.

The simulator platform, Simics, that will be used as a base for this supports loadable objects. Two such objects that will be developed is a memory hierarchy and a timing simulator. The memory hierarchy contains a functional model of the memory system including caches. The timing simulator is fed with a trace of functions being performed and calculated the time for these.

6.2.4 Results

The expected academic results of the project is:

- Modular simulator technology
- A timing analysis tool for a multiprocessor
- Efficient methods for generating and verifying a functional processor model
- Increased knowledge of worst case behaviour of such a system
- Increased knowledge of performance characteristics of a commercial application

Ericsson UABs outcome of the project is mainly increased knowledge about the performance of their application.

- Performance analysis of a database system.
- A tool to develop and optimize applications on.
- Increased knowledge in performance measurement and optimization

6.3 University of Karlskrona/Ronneby

6.3.1 Objectives and Justification

Ericsson Software Technology AB builds parallel applications for collecting information about calls from mobile phones. In such applications, called billing applications, the length and cost of each individual call must be recorded. As a result, a large amount of data must be collected and processed during busy hours. A key component in billing applications is the billing gateway system, which is responsible for the actual collecting and processing of the call data, e.g., sorting of billable and not billable calls. This gateway system can easily become a performance bottleneck if not carefully designed.

Today, the gateway application is written using Solaris threads. The experience using threads is still quite limited and a lot as issues need to be addressed. Examples of such issues are: the granularity of each thread, static vs. dynamic thread creation/deletion, and the trade-offs between traditional processes and light-weight threads. In an initial study, very limited speedup was found in the application when the number of processors exceeded four.

The focus of the work at University of Karlskrona/Ronneby and Ericsson Software Technology AB will be on how to use SMPs for billing applications in mobile phone systems. The objective is to provide *performance prediction methods and guidelines/methodologies for designing efficient parallel applications for SMPs*. The technology transfer to Ericsson Software Technology AB will be experiences, methodologies, and tools to utilize SMPs in their application domain. In order to achieve these goals, the following subprojects will be carried out:

- Characterization of the interaction between the application, the operating system, and the memory system;
- Development of performance prediction methodologies;
- Development of methodologies, techniques, and tools to guide the programmer to better utilize SMPs.

6.3.2 Approach

In order to understand the complex interaction between the hardware and software (both application and system software) it is important to monitor various aspects of how the parallel application behaves, e.g., the communication and synchronization structure of the application, and the memory system behavior. The objective of the first subproject is to characterize this interaction, and armed with this knowledge design applications that better utilize the currently available and also future multiprocessor hardware and operating systems. We expect that we will be able to generalize our knowledge into a set of guidelines for designing parallel applications. Our approach to characterize the application behavior is to do measurements on large billing gateways applications on existing hardware, i.e., case studies of real-world industrial configurations. The operating system we intend to use is Solaris, and we consider acquiring the source code for Solaris in order to better understand the interaction between the application and the underlaying operating system and hardware.

The goal of the second subproject is to develop performance prediction methodologies and tools. Our approach is to collect run-time information when a parallel/multithreaded application is executed on a uniprocessor. We collect information about, e.g., the synchronization behavior, the creation and temination of threads, and changes in the execution state. Important aspects to address in this subproject is the accuracy in the predictions, the collection overhead, and the possible intrusion and change of the application behavior.

In the third subproject the guidance of the application programmer is of primary concern. The goal is to develop tools and methodologies to help the programmer efficiently utilize SMPs. As an example, a very early prototype of a tool to record and graphically visualize the dynamic creation and deletion of threads, and also the synchronization behavior between the threads has been developed. The prototype tool currently has several major limitations, e.g., it does not capture the I/O behavior of an application. Within this subproject, we plan to extend the functionality of the tool to cover a large class of parallel applications.

In the intersection between the three subprojects lies the need for techniques and tools to present measured and simulated results, e.g., support for visualization of both program behavior and simulation statistics. Important aspects are also tools and techniques for 'condensation' of execution and simulation data in order to quickly grasp through large amount of statistics.

Finally, within our project we have an SMP with 8 processors that we will use as: a test bench to get the big picture of, e.g., the synchronization behavior of a parallel program; a platform for operating system experiments; a platform to run (large and long running) simulations on; a platform for parallel program development; and finally for validation of speed-up prediction methods.

6.3.3 Results

The expected academic result from this subproject are

- Performance characterization of the (non-trivial) interaction between the memory system, the system software, and the parallel application. Based on this knowledge we intend to develop a set of guidelines on how to design high-performance parallel applications, e.g., guidelines concerning the tradeoff between using traditional operating system processes or threads when implementing parallel execution.
- Tools and methodologies to predict the performance of parallel applications for SMPs.
- Tools and methodologies to support the programmer in his task of designing efficient parallel applications for SMPs.

Consequently, we expect that we will be able to develop guidelines which can be used in the early design phases as well as tools and methodologies which can be used by the programmer during the implementation phase.

The major contribution by Ericsson Software Technology AB will be to provide the parallel gateway application and application domain knowledge. The expected outcome of this project for Ericsson Software Technology AB are techniques and methodologies to answer questions like

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

6.4 Uppsala

6.4.1 Objectives and Justification

The main objective of PAMP is to develop methods and tools for exploiting symmetric multiprocessors for real-time applications. Uppsala University will address the communication, parallel protocol processing and network interface issues in the design of SMPs.

With the rapid growth of networking and networked applications, the demands put on the communication subsystem have increased tremendously the past years and will continue to increase. Since the communication performance bottleneck in today’s high-speed networks most often is found in the nodes rather than in the physical links themselves, methods that better utilize the available systems are needed. We foresee a growing number of SMP platforms both as end-node clients and as network servers. Hence, there is a need for methods and tools for the efficient implementation of communication protocols on SMP systems. One class of networked applications that is increasing is applications with Quality-of-Service (QoS) demands, e.g. the increasing number of multimedia applications. Communication subsystem resource management must therefore be able to meet these QoS demands.

The Communication and Distributed Systems group at the Department of Computer Systems, Uppsala University, has a research competence within multiprocessor protocol implementations. The group has been working in the data communication area for more than a decade with a focus on end system (host) issues. It has produced prototypes of multiprocessor implementations of protocols, efficient network interfaces and measurement tools. Members of the group have also contributed to Ericsson's switch control protocols. There are several relevant projects ongoing at Uppsala University of which PAMP will benefit from. They include the Esprit Long Term Research project HIPPARCH, on new communication protocol architectures, and a project for Norwegian Telecom Research on a network multiprocessor server interface.This interface is between an internal SCI network and an external, high rate ATM network. Other relevant work include multiprocessor implementations of the Mach operating system and Desk Area Networks for multi computer systems.

The focus of the Uppsala work will be on support for parallel protocol implementation on SMPs and on operating system/run time environment support for protocols written in high level languages such as Erlang. The industrial partner is Ericsson UAB and the contact person is Håkan Millroth/Bjarne Däcker.

The relation between Uppsala and the other partners could be described in terms of exports/imports. Uppsala will export the experience we have on run-time systems for protocols to Ericsson UAB/Erlang and thread handling to University of Ronneby. Our work on SCI/ATM interfaces should be of interest for Ericsson. Recent work on resource reservation for real time communication in the OS kernel seems to be relevant for Chalmers. In terms of importing, Uppsala will get knowledge from Chalmers/SICS/Ronneby on SMPs in general and in particular on simulation methods, tools and environments for SMPs.

6.4.2 Approach

The proposed work will be divided into two subtasks, one on network interfaces and one on operating systems support for communication.

Multiprocessor network interfaces: A network interface to a multiprocessor is different to a uniprocessor in some fundamental ways. The first issue is where it should be located? It may be shared by all processors or attached to one of them or there may be multiple interfaces attached to several processors which then need to be synchronized.

The second issue is how communication data is distributed to the processors from and to the network interface(s). A multiprocessor has an internal interconnect for distribution but it may then become be the performance bottleneck in the distribution of data to the appropriate processors.

A third issue is how protocol processing of higher layers can be distributed and parallelized over the processors.

In our approach we intend to write simulation models of different network architectures that are detailed enough to get predictive performance results. The work builds on previous research on ATM/SCI interfaces, ATM/TCP interaction, the design of efficient network APIs (sockets) and parallel protocol implementations.

Phase 1 of this subtask will include the characterization of those aspects of multiprocessor networking that must be modelled in order to correctly capture the behaviour of the system with respect to performance.

Phase 2 of this subtask will include the development of simulation models that capture the aspects characterized in phase 1 of the subtask.

Phase 3 of this subtask will be an analysis of the predictive capabilities of the developed models and the applicability to networked applications.

Real-time communication support: The internet of the future will be very heterogeneous, ranging from optical based high capacity network to wireless networks with low bandwidth, high error rate, and varying connectivity. Higher layer protocols and distributed applications are likely to be adaptive to compensate for this heterogenity. Another trend is the use of “fat” network servers as proxies or agents to carry out parts of the tasks of “thin” clients with low capacity or poor network connectivity. Many applications of the future will require real-time and predictive services. Examples of such applications include video conferences, virtual meeting rooms, distributed interactive simulations and plain old telephony. In order to meet the Quality of Services (QoS) required by these applications, resources in the network, in network servers, as well as in the end systems must be reserved and scheduled according end-to-end QoS parameters.

We are doing research on operating systems that support real-time applications and higher layer protocols. The dominant resources in a uniprocessor are the CPU, the primary memory and the bus/network interface which must be scheduled in a concerted way. Their interaction between and the reservation schemes of the network, (such as ATM traffic classes or the RSVP and Intserv styles of the Internet) are currently poorly understood. Furthermore, this interaction complexity is much higher in a multiprocessor environment with multiple resources of each class.

In this subtask we will study how the operating system can support adaptive protocols and real-time applications written in high level languages, such as Erlang. It is an extension to the scheduling work at Chalmers with respect to communication. We will investigate operating system mechanisms and abstractions for the programmers to control the resources of the machine. Current general purpose operating systems such as Unix, do not offer sufficient control or have inefficient mechanisms for real-time distributed applications. In our approach, we will study how new operating system nano-kernels that exports “lighter” mechanisms and abstractions for user access to kernel resources will work with the Erlang Run Time system. In particular, we will study the kernel Exokernel from MIT.

Phase 1 of this subtask includes the identification of the resources that must be controlled by the operating system in order to be able to give Quality-of-Service guarantees.

Phase 2 of this subtask includes the development of mechanisms and policies for communication subsystem resource management in an SMP environment.

Phase 3 of this subtask includes the application of resource management as developed in phase 2 of this subtask to applications with Quality-of-Service demands.

6.4.3 Results

We expect a number of academic and industrial results regarding protocol processing and network interfaces to SMPs. The expected academic results are:

- Methods for performance characterization of high performance communication subsystems for SMPs.
- Resource reservation strategies for real-time protocols in an SMP.

The expected industrial results include:

- A kernel and a Run Time Support System interface for protocols written in Erlang with a fine grained control of host and network resources. This result is of interest for Ericsson UAB.
- A simulation platform on which various protocol processing strategies could be modelled.
- Partitioning and parallelizing strategies for multiprocessor protocol implementations, which are anticipated to be of interest for Ericsson UAB.

7. Budget

The initial planning of the program resulted in about 10 Ph. D. project proposals. Based on the general funding principles that has been developed within ARTES, and other minor program costs in terms of administration and mobility actions, the ambition of the program is to generate 8 Ph. Ds and 2 Lic. degrees. The detailed budget for the program is provided in the ARTES program plan.

8. References

8.1 Chalmers

P. Stenström and F. Dahlgren (eds). Applications for Shared-Memory Multiprocessors. Special Issue, *IEEE Computer* Dec 1996.

P. Stenström, M. Brorsson, F. Dahlgren, H. Grahn, and M. Dubois. Boosting Performance of Shared-Memory Multiprocessors, *IEEE Computer*, July 1997.

P. Stenström, E. Hagersten, D. Lilja, M. Martonosi, and M. Venogupal. Trends in Shared-Memory Multiprocessing. *IEEE Computer*, December 1997.

J. Jonsson. The Impact of Application and Architecture Properties on Real-Time Multiprocessor Scheduling, PhD thesis, School of Electrical and Computer Engineering, Chalmers University of Technology, Göteborg, Sweden, Augusti 1997.

J. Jonsson and K. G. Shin. A Parametrized Branch-and-Bound Strategy for Scheduling Precedence-Constrained Tasks on a Multiprocessor System, In *Proceedings of the 26th Int'l Conference on Parallel Processing*, Bloomingdale, Illinois, USA, August 11--15, 1997, pp. 158--165.

J. Jonsson and K. G. Shin, Deadline Assignment in Distributed Hard Real-Time Systems with Relaxed Locality Constraints. In Proceedings of the *17th IEEE Int'l Conference on Distributed Computing Systems*, Baltimore, Maryland, USA, May 27--30, 1997, pp. 432--440.

T. Möller, Fast Bitmap Stretching, in David Kirk (editor), *Graphics Gems III*, chapter 1.1, page 3-7, Professional, Boston, 1995

T. Möller, Faster Ray Tracing Using Scanline Rejection, in Alan W. Paeth (editor), *Graphics Gems V*, chapter 5.3, page

242-257, AP Professional, Boston, 1995.

T. Möller, Radiosity Techniques for Virtual Reality-Faster Reconstruction and Support for Levels of Detail. in Proceedings of the *Fourth International Conference in Central Europe on Computer Graphics and Visualization'96*, page 209-216, Plzen, Czech Republic 1996.

8.2 SICS

P. Magnusson: "A Design for Efficient Simulation of a Multiprocessor", in *Proceedings of MASCOTS*, pages 69-78, January 1993

P. Magnusson and D. Samuelsson: "A Compact Intermediate Format for SIMICS", *Technical Report T94:17, Swedish Institute of Computer Science*, September 1994.

D. Samuelsson: "System Level Interpretation of the SPARC V8 Instruction Set Architecture", *Technical Report R94:23, Swedish Institute of Computer Science*, August 1994.

P. Magnusson and B. Werner: "Efficient Memory Simulation in SimICS", in *28th Annual Simulation Symposium*, April 1995

P. Magnusson, Peter S, Fredrik Dahlgren, Håkan Grahn, Magnus Karlsson, Fredrik Larsson, Fredrik Lundholm, Andreas Moestedt, Jim Nilsson, Per Stenström, Bengt Werner. 1998. SimICS/sun4m: A Virtual Workstation. *In Proceedings of the 1998 USENIX Annual Technical Conference (USENIX'98)*. June 15-19.

Magnusson, Peter S. 1997. Efficient instruction cache simulation and execution profiling with a threaded-code interpreter. In *Proceedings of the Winter Simulation Conference (WSC'97)*. December 7-9.

Montelius, J. and P. S. Magnusson. 1997. Using SimICS to evaluate the Penny system. In *Proceedings of ILPS'97*.

8.3 Karlskrona/Ronneby

L. Lundberg and H. Lennerstad, "An Optimal Upper Bound on the Minimal Completion Time in Distributed Supercomputing," in *Proc. 1994 International Conference on Supercomputing*, Juli 1994, Manchester, England.

H. Lennerstad and L. Lundberg, "An Optimal Execution Time Estimate of Static versus Dynamic Allocation in Multiprocessor Systems," *SIAM Journal of Computing*, Vol 24(4), pp. 751-764, August 1995.

L. Lundberg, "Multiprocessor Performance Evaluation of Billing Gateway Systems for Telecommunication Applications," in *Proc. ISCA International Conference on Parallel and Distributed Computing Systems*, Dijon, France, September 1996.

H. Grahn and P. Stenström, ``Evaluation of a Competitive-Update Cache Coherence Protocol with Migratory Data Detection,'' in *Journal of Parallel and Distributed Computing*, 39(2):168-180 (December 1996).

H. Grahn, P. Stenström, and Michel Dubois, ``Implementation and Evaluation of Update-Based Cache Protocols Under Relaxed Memory Consistency Models,'' in *Future Generation Computer Systems*, 11(3):247-271 (June 1995).

H. Grahn and P. Stenström, "Efficient Strategies for Software-Only Directory Protocols in Shared-Memory Multiprocessors," in *Proc. of 22nd International Symposium on Computer Architecture*, pp. 38-47, June 1995.

M. Broberg, L. Lundberg, and H. Grahn, "VPPB - A Visualization and Performance Prediction Tool for Multithreaded Solaris Programs," in *Proc. 12th Int'l Parallel Processing Symposium*, March-April 1998 (to appear).

Håkan Grahn and Per Stenström, ``Evaluation of a Competitive-Update Cache Coherence Protocol with Migratory Data Detection,'' in Journal of Parallel and Distributed Computing, 39(2):168-180 (December 1996).

Håkan Grahn, Per Stenström, and Michel Dubois, ``Implementation and Evaluation of Update-Based Cache

Protocols Under Relaxed Memory Consistency Models,’’ in Future Generation Computer Systems, 11(3):247-271 (June 1995).

8.4 Uppsala

Björkman, M. & Gunningberg, P., “Performance Modeling of Multiprocessor Implementations of Protocols”. Accepted for publication in the ACM /IEEE Transactions on Networking.

Ahlgren, B. Björkman, M. & Gunningberg, P., “The Applicability of Integrated Layer Processing”. Accepted for publication in Journal of Selected Areas of Communication during 1998.

Ahlgren, B. Björkman, M. & Gunningberg, P., “Integrated Layer Processing can be hazardous to your performance”. IFIP Workshop on protocols for High Speed Networks V, October 1996, Chapman & Hall.

Ahlgren, B. Björkman, M. & Gunningberg, P., “Towards predictable ILP Performance - Controlling Communication Buffer Cache Effects”. Australian Computer Journal, May 1996.

Gunningberg, P. & Kure, O, “ATM as a memory interconnect in a Desk Area Network”. IFIP High Performance Networking 1995, Kluwer, September 1995.

Moldeklev, K. & Gunningberg, P., “How a big ATM MTU causes deadlocks in TCP data transfers”, ACM /IEEE Transactions on Networking, August 1995, Vol. 3, No 4.

Ahlgren, Gunningberg, P. & Moldeklev, K, “Increasing Communication Performance with a Minimal-Copy Data Path Supporting ILP and ALF, Journal of High Speed Networks, 5, 1996.

Björkman, M. & Gunningberg, P., “Locking Effects in Multiprocessor Implementation of Protocols”. Journal of High Speed Networks, 1994, No 2, Vol 3.