ARTIST 2

Network of Excellence

IST-004527 ARTIST2:
Embedded Systems Design

Activity Progress Report for Year 1

JPIA-Platform:

A Common Infrastructure for Adaptive Real-time Systems

Cluster:

Adaptive Real-Time

Activity Leader:

Giorgio Buttazzo (University of Pavia)

The objective is to show how current operating systems and network protocols have to be extended to support emerging real-time applications that exhibit a high degree of complexity and operate in dynamic environments.

The impact on operating system standards (like RT-POSIX and OSEK) as well as network protocols will also be taken into account.

Table of Contents

1. Introduction

1.1 Activity Leader

Team Leader: Giorgio Buttazzo (Univ. of Pavia)
Areas of his team's expertise: real-time scheduling, robotic systems and interfacing.

1.2 Policy Objective

The objective is to show how current operating systems and network protocols have to be extended to support emerging real-time applications that exhibit a high degree of complexity and operate in dynamic environments.

The impact on operating system standards (like RT-POSIX and OSEK) as well as network protocols will also be taken into account.

1.3 Industrial Sectors

Consumer electronics, multimedia systems, automotive, complex robotics systems. In the near future we plan to contact a number of local automotive industries that are interested in the development of distributed control systems for tractors and agricultural machines.

2. Overview of the Activity

2.1 Artist Participants and roles

Team Leader: Luis Almeida (University of Aveiro)
Areas of his team's expertise: networking platform, development of distributed applications.

Team Leader: Gerhard Fohler (Mälardalen University)
Areas of his team's expertise: video streaming, scheduling.

Team Leader: Michael Gonzalez Harbour (University of Cantabria)
Areas of his team's expertise: definition of the POSIX operating system interface.

Team Leader: Alan Burns (University of York)
Areas of his team's expertise: Real-Time Java.

Team Leader: Juan Antonio De la Puente (UP Madrid)
Areas of his team's expertise: Quality of Service management architecture and tools.

Team Leader: Eduardo Tovar (Polytechnic Institute of Porto)
Areas of his team's expertise: distributed applications and QoS over heterogeneous networks.

2.2 Affiliated partners and Roles

Team Leader: Giuseppe Lipari (Scuola Superiore S. Anna)
Areas of his team's expertise: kernel maintenance.

Team Leader: Paolo Gai (Evidence s.r.l.)
Areas of his team's expertise: real-time kernels and OSEK standard.

Team Leader: Pau Marti (Universitat Politècnica de Catalunya)
Areas of his team's expertise: control methodologies.

Team Leader: Carlos Delgado (U. Carlos III of Madrid)
Areas of his team's expertise: QoS Management.

2.3 Starting date, and expected ending date

Starting date: September 1st, 2004

Expected ending date. September 2006, when all the partners have developed and tested their real-time applications using the Shark operating system and become familiar with the kernel internals.

2.4 Baseline

The teams working in this activity are bringing significant pieces to this platform:

University of Pavia, Scuola Superiore S. Anna, University of Aveiro, Mälardalen University, and affiliated partners Universitat Politècnica de Catalunya and Polytechnic Institute of Porto will share the SHARK real-time operating system (developed by Scuola Superiore S. Anna) for testing novel scheduling algorithms and resource management policies in different application environments and case studies available in those sites. University of Cantabria will provide support for implementing a POSIX interface, and University of York will use it to develop an implementation of the real-time specification for Java, on which new APIS and services for future versions of the specification can be tested. University of Pavia will use SHARK to port real-time components developed for RT-Linux, to make them available to a broader audience. UP Madrid and U. Carlos III of Madrid will evaluate the possibility of implementing their HOLA QoS architecture for quality of service management. Mälardalen University will evaluate the application-defined scheduling API to test novel flexible scheduling algorithms on adaptive multimedia applications, which are already investigated for the consumer electronics domain.

University of Aveiro will use SHARK to implement the FTT (Flexible Time-Triggered) networking framework (developed by University of Aveiro) to support adaptable distributed real-time applications in dynamic environments. This networking framework will be made available to all other partners to develop distributed applications.

The Polytechnic Institute of Porto will provide a distributed platform with wired and wireless segments that supports different traffic types, which will be used for the development of QoS communication support over heterogeneous networks. This involves co-existence of control traffic and multimedia traffic. Moreover, the Polytechnic Institute of Porto will use SHARK to implement the Ethernet/IP framework (undergoing bilateral project with Rockwell automation) to support distributed industrial control applications over switched-Ethernet communications. A test bed will be provided for this.

Affiliated industrial partners Evidence will exploit the results by using flexible scheduling mechanisms for developing competitive commercial kernels for automotive and multimedia applications.

2.5 Technical Description

The work will focus on how to extend current operating systems and network features to support emerging real-time applications that exhibit a high degree of complexity and dynamic behaviour.

The impact on standard operating system interfaces, like RT-POSIX and OSEK will be evaluated to make a concrete proposal for modifying them. Developing real-time applications and components in a POSIX compliant operating system will promote portability to other POSIX platforms and will challenge the current standard to be extended to better meet the needs of advanced applications with flexible scheduling requirements.

In the short term (18-month), the Shark operating system will be installed on all the sites involved in the activity. The work will consist in configuring the kernel and the network for the specific hardware devices and system configuration, and making all the partners capable of using the kernel and network protocol to develop their own real-time software.

3. Activity Progress Report

3.1 Work achieved in the first 6 months

To build a common infrastructure for testing novel scheduling algorithms and resource management policies in different application environments, we had to select a real-time operating system aimed at simplifying the integration work to be done by the different partners. In other words, instead of spending time in knowing the internal details of a particular operating system, we wanted to concentrate on specific kernel mechanisms and on how to replace them with new ones with desired features. To simplify this work, we started seeking for real-time kernels with a modular structure, in which the implementation of a mechanism were as much as possible independent on the implementation of the other parts. In additions, the kernel had to be open source to allow extentions, well documented, and availble for PC-based platforms, supporting a sufficient number of devices to allow partners to develop control applications. Finally, in order to allow the development of algorihtms for overrun and overload management and to compare the performance of different algorithms, we wanted a kernel able to measure the execution of application tasks.

Unfortunately, many open source real-time kernels do not satisfy these requirements. In fact, operating systems like RT-Linux and RTAI have a quite complex structure, and the scheduler has several dependency with the other kernel mechanisms, hence it cannot be isolated and easily replaced with a new one. The only kernel we found suitable for this project was Shark. The reasons for choosing such a kernel are explained below.

1. Shark (Soft and HArd Real-time Kernel) has a modular structure that allows the user to develop new scheduling algorithms and new resource management policies independently of other kernel mechanisms. This feature is essential to speed up the research activities of the ART cluster and cannot be found in today's operating systems.
2. Under Shark, the same application can be tested under different scheduling policies and resource management protocols, without changing the source code. This is achieved by using the concept of task model, which breaks the dependency of the application from a specific scheduler or resource protocol. Moreover, the algorithms can be dynamically selected by the user at system initialization, through a configuration file. This feature cannot be found in today's operating systems.
3. Shark allows the user to combine different scheduling policies into the same system, so providing support for those applications consisting of tasks characterized by different criticality and timing models.
4. Shark provides support for periodic and aperiodic control tasks with explicit timing constraints and automatically handles the activation of periodic tasks.
5. It includes drivers for the most common I/O peripherals.
6. It complies with the POSIX standard, PSE51 profile, so facilitating the porting of a real-time application to different operating systems and platforms.
7. It includes user manuals and several sample real-time applications to help the user in approaching the kernel.
8. Shark was developed at the ReTiS Lab of the Scuola Superiore Sant'Anna of Pisa, in collaboration with the Robotic Laboratory of the University of Pavia, hence the know-how for maintaining and updating the kernel is internal to the ART cluster.

9. Shark is currently adopted in more than 20 universities in Europe, United States, South America and Asia, as a tool for teaching real-time programming, and for the development of control demonstrators based on novel scheduling techniques. This facilitates the activities of the ART cluster and favors the exchange of new algorithms.

The first action carried out was to develop a web site (http://feanor.sssup.it/retis-projects) to create a forum for the various Shark users, in which it is possible to exchange messages, search for questions, etc. The web site also includes a page with web links to the various research groups that are using the kernel for control applications, and a page of *Frequently Asked Questions*, to quickly address the most common problems encountered by the developers.

The main action was to organize a workshop to introduce the Shark kernel to all the partners of the ART cluster, enabling the participants to quickly use the kernel, develop simple real-time applications, and implement novel scheduling algorithms. The workshop was held in Pontedera, Pisa (Italy), at the Scuola Superiore Sant'Anna, from February 28 to March 4, 2005.

3.2 Work achieved in months 6-12

A collaboration was started with the University of Siena at Arezzo, to enable the users at the development of real-time control applications on robot devices that are available in their control laboratory. The Scuola Universitaria Professionalizzante della Svizzera Italiana (SUPSI) was also contacted for porting a code generator for Scilab/Scicos tools toward the Shark platform.
The kernel was maintained by writing and updating the existing documentation, removing bugs, and modifying the internal structure of the scheduler. The **task_endcycle()** system call was modified to make it sensitive to system termination.
In additinon, a graduate thesis was started at the ReTiS Lab for porting the low-level layer of Shark (OsLib) to the L4 microkernel.
The University of Aveiro, in collaboration with the University of Pavia, started investigating a kernel methodology for supporting the hard real-time communication over a dedicated Ethernet network.
Finally, new device drivers were implemented based on Linux technology.

3.3 Difficulties Encountered

The major difficulties were encountered in the development of new device drivers, due to the lack of hardware documentation and due to the complexity of the Linux code.

3.4 Recommendations

Develop a glue code to simplify the porting of Linux drivers.

3.5 Milestones

All the partners were tought on the use of the Shark kernel and started to develop their own control applications.

3.6 Main Funding

Main sources of funding have been:

FIRST – "Flexible Integrated Real-time Systems Technology", IST-2001-32467.

OCERA – "Open Components for Embedded Real-time Applications", IST-2001-35102.

3.7 Indicators for Integration

The expected result is to make the kernel and network protocol available by other users, for testing it in other research laboratories and in different real-time applications environments. The platform will be available to all partners, who will gather usage experience that will drive further development of the platform. The platform will become enhanced with the addition of sophisticated services, either local or remote, such as real-time Java, real-time components, new scheduling policies and Quality of service management. It is also expected that the same activity will also have a significant impact on education, since the kernel and network protocol will be used to explain and develop simple RT applications within existing university courses.

3.8 Evolution

The 18 month objective is to deploy a working platform for experimental RTOS and network development. Each partner will concentrate on a specific application with the aim of evaluating the impact of kernel policies on system performance. Sample real-time applications will include mobile robotics, distributed systems, multimedia processing, and process control systems.
The University of Aveiro, in collaboration with the University of Pavia, will investigate a kernel methodology for supporting hard real-time communication over a dedicated Ethernet network. The communication system will be built upon cheap and common off-the-shelf Ethernet hardware, which definitely limits the overall system cost. While addressing the cost issues, the timing requirements needed by the real-time communication will be fulfilled through a careful protocol implementation, that will be designed to take into account jitter and latency introduced by message buffering, direct memory access and timer programming. The protocol will be implemented as a module of the S.Ha.R.K. real-time kernel, allowing the performance evaluation of real working applications.
Another important work planned by the ART cluster for the next 18 months is to develop a repository to collect all the software developed by the ART partners and make it available on the Shark web site. All software contributions will be tested, documented and published using a common format. This action will allow integrating existing relavant work on scheduling algorithms, feasibility analysis, communication protocols, concurrency control policies, applications, etc. into a uniform structure, that can be used and executed within the shared platform. The repository will also simplify access to research products, having the unique possibility of performing new experiments to validate and compare several methodologies developed in the community.

The long term objective of the activity on common platforms is to cause the evolution of RTOS and networking standards, by developing new concepts.

4. Detailed Technical View

4.1 Brief State of the Art

A research platform for real-time systems is composed of competencies, resources, and tools targeting at the development of control applications with performance and timing requirements.

A shared research platform is essential for experimenting new real-time software technology, including novel scheduling algorithms, resource management techniques, energy-aware policies and overload handling approaches to increase robustness and predictability. Platforms are also used as the basis for transfer research results to industry, as they allow teaching practical knowledge of the concepts and techniques.

The common infrastructure for adaptive Real-time Systems developed within ARTIST2 is a platform that integrates an existing operating system, called Shark, with novel real-time technology resulting from long-term efforts. It is meant to be durable and evolving with the state of the art.

This is possible because Shark was designed to be modular through a componend-based software technology, so that new algorithms can be easily incorporated into the kernel and integrated in a hierarchical fashion, depending on the user needs. The kernel modularity allows all the internal mechanisms, like scheduling and mutual exclusion protocols, to be developed independently of each other, by following an event-driven approach and a predefined programming interface. The user can also replace a given algorithm with a different one, without changing the application code. This is important when testing the behavior of a specific scheduling policy on a particular control application, since it simplifies and speeds up the experimentation phase.

The Shark (Soft and HArd Real-time Kernel) operating system was developed at the ReTiS Lab of the Scuola Superiore Sant'Anna of Pisa, with the collaboration of the Robotic Laboratory of the University of Pavia. It is a very suited tool for disseminating real-time software technologies both at an education and prototyping level, because it provides a number of internal kernel mechanisms specifically designed to facilitate the development of demonstrators and prototypes. The Shark operating system has the following main characteristics:

- it supports applications where computational tasks can have explicit timing constraints;
- it includes several advanced algorithms for task scheduling and shared resource management, which can be dynamically selected by the user through a configuration file.
- it includes drivers for the most common I/O peripherals;
- it complies with the POSIX standard, PSE51 profile;
- it includes user manuals and several sample real-time applications.

Currently, the Shark operating system is adopted in about 20 universities in Europe, United States, South America and Asia, as a tool for teaching real-time programming, and for the development of control demonstrators based on novel scheduling techniques.

4.2 Industrial Needs and Experience

One of the major difficulties for transferring novel real-time kernel technologies into the industrial practice is that commercial operating systems are not flexible enough to incorporate new kernel mechanisms, and complex enough to be modified withouth affecting the other internal components. In addition, existing applications should not be affected by the new kernel features, since changing the application code is expensive and unsafe.

As a consequence, although most industrial developers recognize the limitations of existing tools, the modification of an operating system or the use of a different one, would even create bigger problems in terms of costs, safety, and backward compatibility.

The use of shared platforms would facilitate the transfer of research results to industry, as they allow teaching practical knowledge of the concepts and techniques. In addition, several solutions could be developed and tested in parallel in different partner sites, allowing the evaluation of the most appropriate approach for the specific application.

4.3 Ongoing Work in the Partner Institutions

The first action carried out by the University of Pavia, in collaboration with the affiliates Evidence and the Scuola Superiore Sant'Anna, was to develop a web site (http://feanor.sssup.it/retis-projects) to create a forum for the various Shark users, in which it is possible to exchange messages, search for questions, etc. The web site also includes a page with web links to the various research groups that are using the kernel for control applications, and a page of *Frequently Asked Questions*, to quickly address the most common problems encountered by the developers.

The main activity was to organize a workshop to introduce the Shark kernel to all the partners of the ART cluster, enabling the participants to quickly use the kernel, develop simple real-time applications, and implement novel scheduling algorithms. The workshop was held in Pontedera, Pisa (Italy), at the Scuola Superiore Sant'Anna, from February 28 to March 4, 2005.

During the workshop, participants were asked to develop a small software project using Shark. They were divided into groups and each group had to develop a software module or at least design the structure of the application. The time available for programming was not much, therefore some group developed some simple demo, while some other participant just sketched the idea of a more complete kernel component.

These projects were autonomously continued within each partner institution, and they are briefly described below:

4.3.1 Vision guided mobile robot

The goal was to remotely control a small mobile robot by visual feedback, so that the robot (equipped with a wireless camera) could track an object with a given color. The video camera was connected to a PC running Shark. The software was processing the image at 25 frames per second to compute the position of the target. Depending on this position, the PC sent the commands to the cart through a wireless network. The hardware was kinldy provided by the Robotic Lab of the University of Pavia and most of the drivers and libraries were already available. The participants developed the application by integrating the existing software components in a nice and innovative way.

4.3.2 Selfish Round Robin scheduler

This project was carried out independently by two different groups. The goal was to implement the Selfish Round Robin scheduler into Shark. The participants had to develop a new scheduling module and try its properties by running some of the Shark demos. Both groups were able to complete the implementation in only two days!

4.3.3 Hierarchical scheduler

Another group investigated the possibility of implementing a hierarchical scheduling system on top of Shark, with capacity sharing.

4.3.4 ADA programming interface

Another group of two students presented the design of an interface module that would allow porting the Ada run-time system on Shark, so enabling the kernel to execute real-time applications in Ada.

4.3.5 Error-based scheduling

Another group proposed the design of a new scheduling strategy that tries to minimize the overall system "scheduling error". Given a definition of "error" for a control task, the idea is to design a scheduler that should minimize the overall system error, when the system is overloaded. The group presented the scheduler design and discussed possible implementations.

Finally, the use of the Shark operating system was also extended to a partner of the Control cluster. To do that, Dr. Anton Cervin, from University of Lund, was invited to spend 6 months at the University of Pavia, to work on adaptive real-time scheduling techniques to be implemented into Shark. During this period, he learned how to use the kernel and how to develop new scheduling algorithms. This experience is now being used at Lund to develop real-time control applications based on innovative scheduling algorithms. The collaboration between the ART cluster and the Control cluster is continuing to investigate how the scheduler can be designed to reduce the effects of delays and jitter in control applications.

4.4 Interaction, Building Excellence Between Partners

The interaction among the ART partners within Artist2 is helping to integrate a lot of existing work (scheduling algorithms, feasibility analysis, communication protocols, concurrency control policies, etc.) into a common infrastructure. When this is accomplished, each partner will concretely access the products of research, having the unique possibility of performing new experiments to validate and compare several methodologies developed in the community. This is a crucial condition for building excellence within the team of the cluster.

4.5 Spreading Excellence

The expertise built in the ART cluster using the common infrastructure for adaptive real-time systems will be used to spred excellence towards

Industry: contacts with industries will be established to show the results of the activity and illustrate the benefits of the novel kernel technologies. Industries more interested in these results are those involved in automotive, consumer electronics, telecommunications, and robot control.

Education: the real-time kernel infrastructure set up for this activity will be used to organize courses, workshops and summer schools, for teaching how to apply theory into practice and to concretely illustrate the effects of kernel policies on the behavior of time sensitive applications.

In addition, the ART cluster will develop a repository to collect all the software developed by the ART partners and make it available to all clusters and to the entire world. All software contributions will be tested, documented and published using a common format. This action will allow integrating existing relavant work on scheduling algorithms, feasibility analysis, communication protocols, concurrency control policies, applications, etc. into a uniform structure, that can be used and executed within the shared platform. The repository will also simplify access to research products, having the unique possibility of performing new experiments to validate and compare several methodologies developed in the community. This is a crucial condition for building excellence within the ARTIST2 Network.