

Preview

This document is a preview version and not necessarily identical with the original.

Framework for Distributed Simulation and Measuring of Complex Relations of Operating System Components

Winfried Kalfa, Volker Fickert, Thomas Weise
Technical University of Chemnitz, department of computer science, operating systems group,
Straße der Nationen 62, 09107 Chemnitz, Germany
Email: kalfa@informatik.tu-chemnitz.de

Abstract: Collecting experience with the help of experiments makes it much easier to understand complex relations. But with modern computing systems it is hardly possible to get a direct insight into internal processes. A current project for the simulation and realization of measuring of processes in operating systems is presented in my lecture. Its main point is the visualization of such processes. Because of that distributed applications which simulate different components of an operating system or which, with the help of appropriate drivers, are able to carry out measurements on real machines were drawn up with JAVA. During the simulation and measurements learners can influence parameters of different components and experience their effects directly. Thus in teaching operating systems it will be possible to illustrate processes which are difficult to understand and to influence complex relations immediately.

Introduction

Experiences from other projects have shown that animations, simulations as well as virtual practical training have a special meaning in teaching operating systems. More complex processes like the resource management in real systems cannot be completely didactically explained because only a theoretical explanation is possible. And so e-learning including interactive elements offers a meaningful and didactic extension of normal teaching allowing the students to experiment on theoretic descriptions.

In former projects a large number of different animations and simulations have been developed. The animations themselves were realized with SVG (Scalable Vector Graphics) and the simulations with JAVA. Thus simulators of processor scheduling, memory management and file system management have been created. These are separate applications which cannot be used together. Our new project starts exactly at this point by extending single simulations so that they can either be used as a single application or form a complex simulation (Fig. 1).

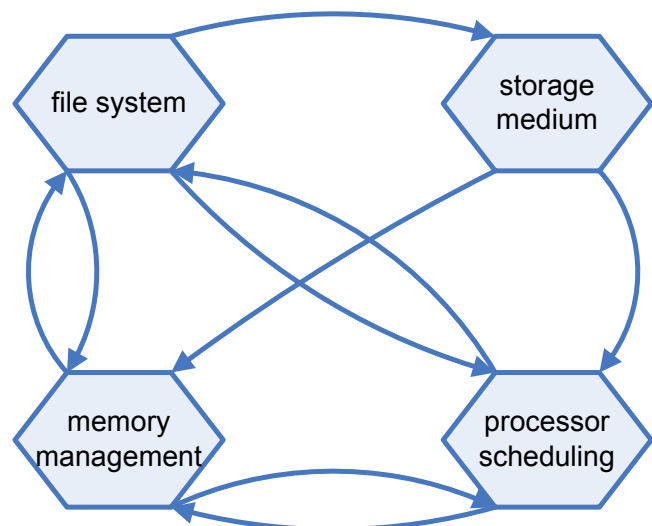


Figure 1: Components of simulation

As the communication of single components is carried out via SOAP (Simple Object Access Protocol), each of them can happen on a computer of its own. From our experience up to now this also makes sense as already one single simulation can overload the latest PC. Besides this design also allows the simple exchange of single simulation tools with drivers, which carry out measurements on real machines.

The services of components can be called up separately by a graphical user interface or with the help of a script resembling the programming language C. In addition the running of components can also be carried out by a configurable probabilistic model. Then the learners get the chance of measuring running processes intensively and evaluating them graphically. The range and the complexity of experiments and measurements can be adjusted to each learner's standard of knowledge.

Communication between the components

In order to simulate a computing system it is divided up into its functional components. It is possible to distribute the single components and so the need of resources on different computers. These computers are linked via a LAN or the Internet. In order to make the simulation of the cooperation of several components possible, these components must exchange respective dates (e.g. for service calls and results) among each other. Here we have used the widespread protocol SOAP (Fig. 2). It supplies us with a simple and transparent mechanism of exchanging structured information. By this the computers, among which information exchange is to take place, can be located in a decentralized, distributed environment. SOAP codes request and result with the simple data protocol XML. For the exchange of data the proven protocol HTTP is used. As SOAP is independent of operating systems, programming languages and object models it is able to connect different platforms and is easy to implement. Further advantages are standardization, openness, robustness and scalability. That is why it is not difficult to extend the given system with new devices or different interpretations.

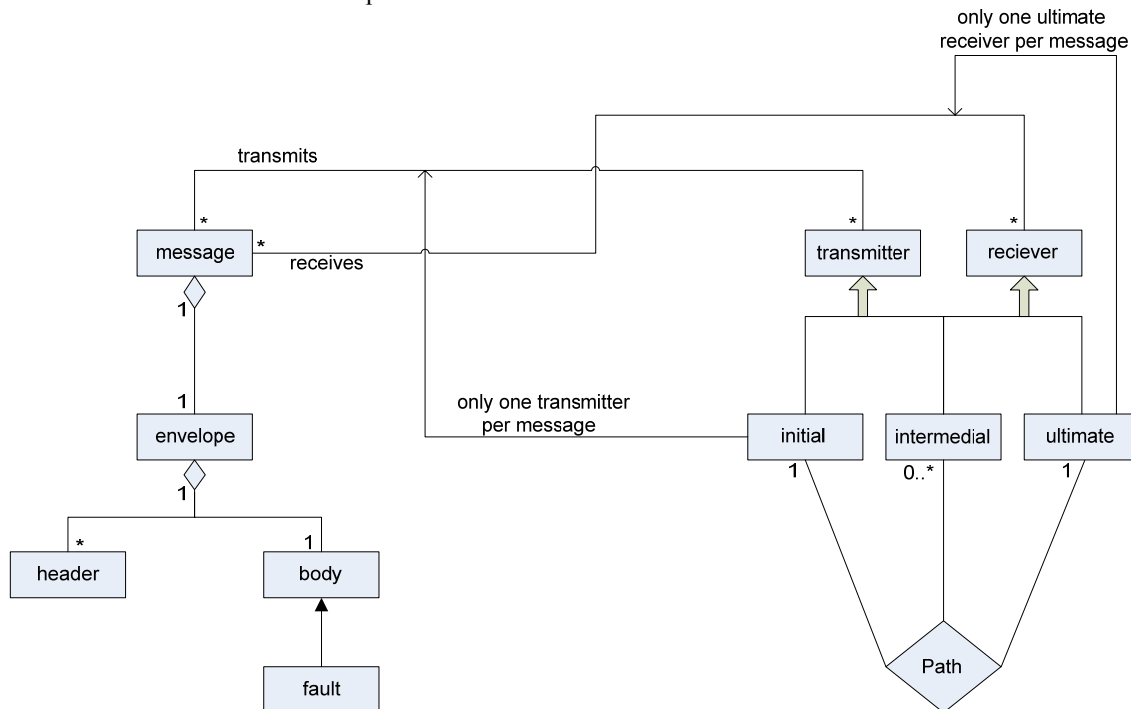


Figure 2: UML (Unified Modelling Language) diagram of SOAP service call

Graphical user interface

The graphical user interface is a central entrance point for the usage of our framework. At the beginning of a simulation the configuration data are transferred to the single components. Then the control of the simulation can be carried out with the help of a script resembling C or by the configuration of a stochastic model.

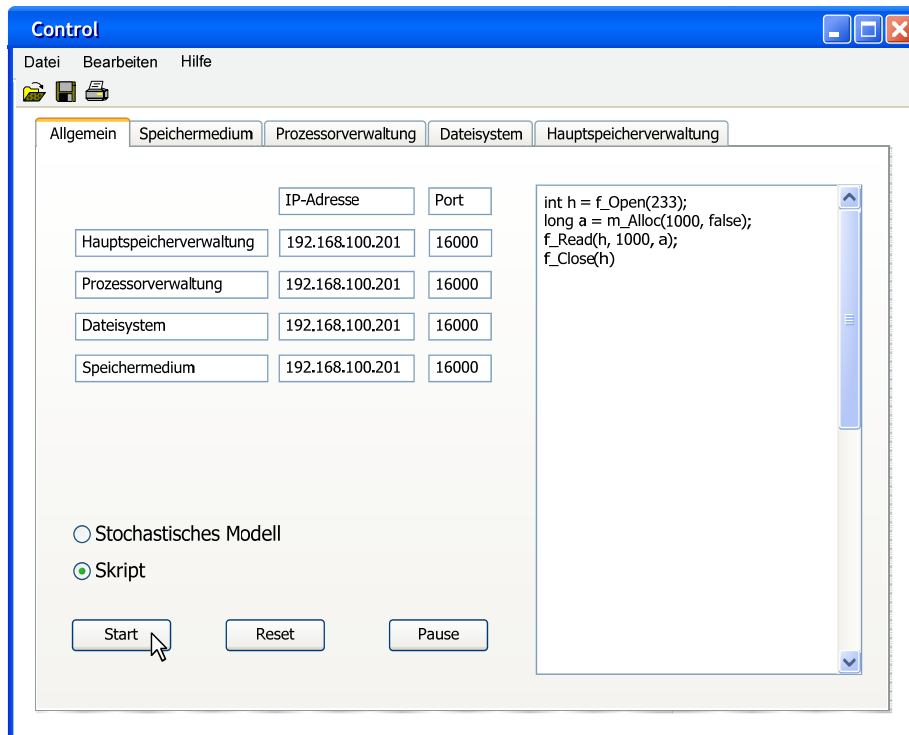


Figure 3: Graphical user interface

Components

The framework consists of the following components: storage medium, memory management, processor scheduling and file system management. Now let us have a closer look at them. In addition to these simulation components there are components for generating load and the evaluation of experiments by numbers and graphics.

Memory Management

The component “memory management” of the computer system simulation includes numerous configuration options. At first its size can be given. But also hardware oriented features like latency period and clock frequency can be configured. Furthermore kinds of addressing like absolute or relative storage are taken into consideration. It can be regulated if the processes running in the computer system are given virtual storage or if a swap file should be used on request.

Besides its complex configuration options memory control offers the following services:

- assigning memory space,
- releasing memory space,
- writing data into memory,
- reading data from the memory, and
- querying memory still available.

As a whole the component “memory management” can be applied to simulate various (memory) environments like a simple embedded system as in coffee machines and watches as well as more complex micro controller systems and the standard PC.

Storage medium

In computer systems different devices are used for long-term storage of data. These are shaped in the “storage medium” component in order to use them for the file system. Different features of different storage media e.g. hard disks, optical drives, tape drives and RAM-disks are simulated. It is absolutely necessary to differentiate between different devices, because all storage media contain different features.

E.g. a hard disk has got different average access times resulting from the arm movements of the read/write head. These are not included in a RAM-drive which uses electronic storage. The total storage capacity with optical drives is standardized, for example, with CD-R 650 megabytes or 700 megabytes can be laid down. The numerous parameters of all storage media are separately adjustable. A virtual disk is a logical resource which provides the services of

- reading a physical record, and
- writing a physical record.

So the features of different storage media are abstracted from and the functionality is standardized.

File system

Computer systems which are used for data processing always support file systems on a so-called “virtual disk”. As a rule such data systems consist of these parts: registry, allocation unit and free space management, which administrate a data carrier separated in clusters. For each of these units of the file system different algorithms can be chosen. Hereby known file systems like FAT and ext2 can be simulated but also completely independent types can be generated.

Services provided by the file system correspond to the known file operations from many programming languages:

- create file,
- open file,
- close file,
- delete file,
- read data from the file or write data into a file,
- move read/write pointer freely, and
- realize size of the file.

Processor scheduling

In modern computer systems many processes run at the same time. Then these must be distributed on a few processors. Different criteria can be used to decide which processor gets which process. This is called processor scheduling. The assigning of processors can happen to various algorithms. The component of processor scheduling comprises besides scheduling also the configuration of features of processors. There is the chance of establishing one or more processors in a system. For this the principle of architecture “Symmetric Multiprocessor System” (SMP) is used. The clock frequency of each processor is adjustable.

The following services are provided:

- create process,
- terminate process,
- make system call,
- request waiting time in seconds, and
- request calculating time in clocks.

Summary

The simulation core has already been completed, the load is still being generated by a separate JAVA program and the output of simulation data takes place on the console. So the operating system framework is not ready for teaching

yet. To realize our project the development of a front end (finished by October 2005) and an extended test of the framework will be necessary. The front end will consist of following components:

1. a central control program,
2. an option of choosing parameters for all parts of the simulation core (Fig. 4),
3. a repeated creation of load,
 - a. by an interpreter similar to C,
 - b. by a stochastic model,
 - c. by measurements of load on real machines,
4. a component for distribution of different components in a LAN with load balancing, and
5. a numerical and graphic presentation of results.

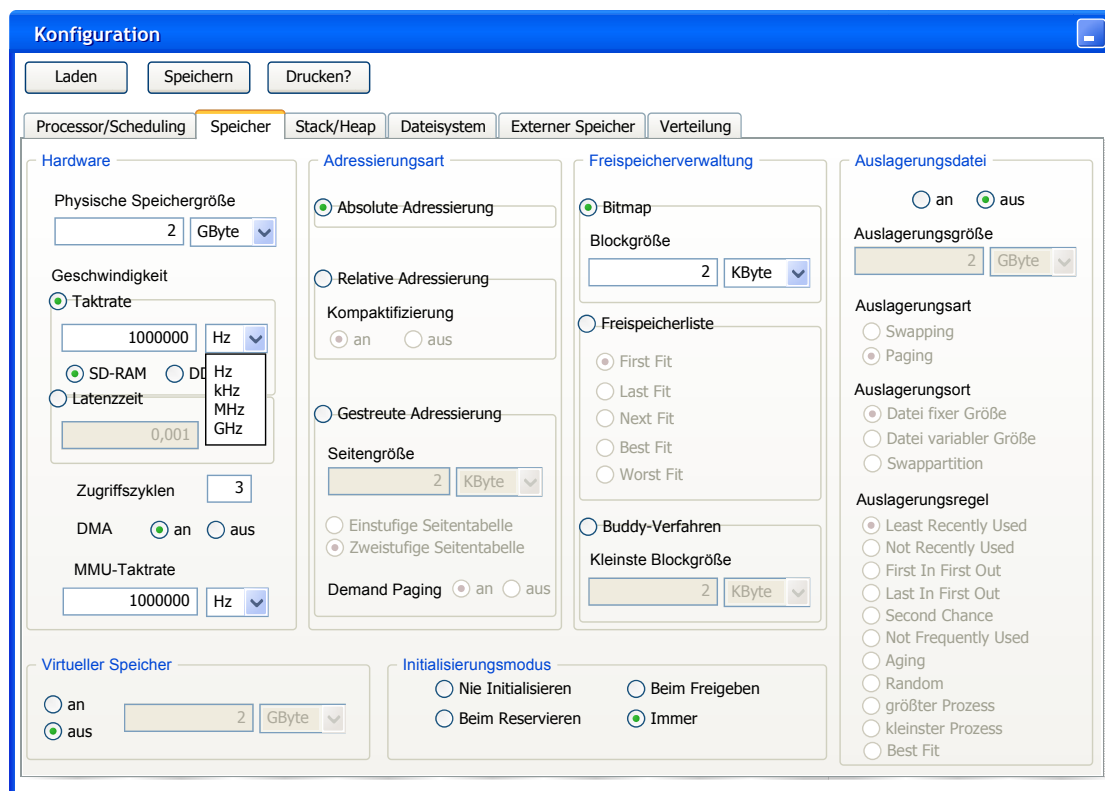


Figure 4: Input of parameters for the simulation core component “memory management”

At the conference the current state of realization will be presented. The use of simulations created and finished up to now in our teachings already shows that students deepen their theoretical knowledge by dealing with the subject interactively and have better learning results.

In the future the partial exchange of single operating systems components in real systems is possible. As a matter of principle, extension by other simulating components, measurements on real machines or the use of other operating systems are possible, too. The sustainability of the project is guaranteed by its modular concept, the use of standardized protocols, and the freely accessible programming language JAVA for the central components.

References

Kalfa, W. (1997), *Praktische Übungen und Experimente in Betriebssystemen*, TU Chemnitz, Betriebssysteme, TR 3.

Koehler, F., Kroeger, R. and Kalfa, W. (2002), *Integrating Simulators and Real-Life Experiments into an XML-based Teaching and Learning Platform*, ED-Media 2002, Denver (co).

Lucke, U. & Tavangarian, D., *Turning a Current Trend into a Valuable Instrument: Multidimensional Educational Multimedia based on XML*, ED-Media 2002, Denver (co).

Wissenswerkstatt Rechnersysteme, (<http://www.wwr-projekt.de>).

Framework for Distributed Simulation and Measuring of Complex Relations of Operating System Components

Winfried Kalfa, Volker Fickert, Thomas Weise
Technical University of Chemnitz, department of computer science, operating systems group,
Straße der Nationen 62, 09107 Chemnitz, Germany
Email: kalfa@informatik.tu-chemnitz.de

Abstract: Collecting experience with the help of experiments makes it much easier to understand complex relations. But with modern computing systems it is hardly possible to get a direct insight into internal processes. A current project for the simulation and realization of measuring of processes in operating systems is presented in my lecture. Its main point is the visualization of processes in operating systems. Because of that distributed applications which simulate different components of an operating system or which, with the help of appropriate drivers, are able to carry out measurements on real machines were drawn up with JAVA. During the simulation and measurements learners can influence parameters of different components and experience their effects directly. Thus in teaching operating systems it will be possible to illustrate processes which are difficult to understand and to influence complex relations immediately.

Introduction

Experiences from other projects have shown that animations, simulations as well as virtual practical training have a special meaning in teaching operating systems. More complex processes like the resource management in real systems cannot be completely didactically explained because only a theoretical explanation is possible. And so e-learning including interactive elements offers a meaningful and didactic extension of normal teaching allowing the students to experiment on theoretic descriptions.

In former projects a large amount of different animations and simulations have been developed. The animations themselves were realized with SVG and the simulations with JAVA. Thus simulators of processor scheduling, memory management and file system management have been created. These are separated applications which cannot be used together. Our new project starts exactly at this point by extending single simulations so that they can either be used as a single application or form a complex simulation (Fig. 1).

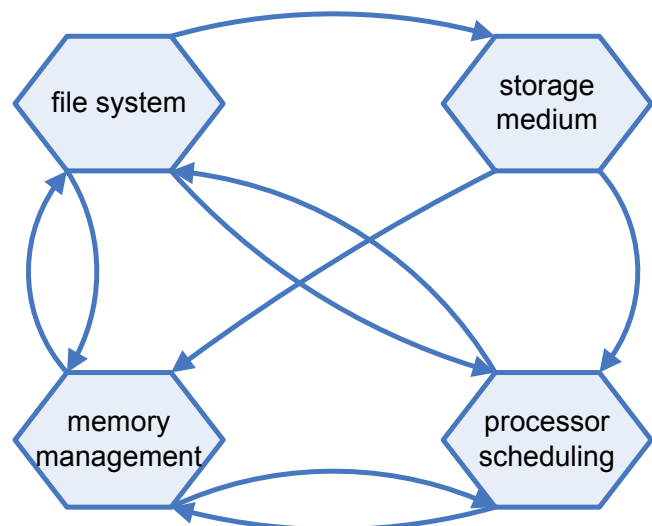


Figure 1: Components of simulation

As the communication of single components is carried out via SOAP (Simple Object Access Protocol), each of them can happen on a computer of its own. From our experience up to now this also makes sense as already on single simulation can overload the latest PC. Besides this design also allows the simple exchange of single simulation tools with drivers, which carry out measurements on a real machines.

The services of components can be called up separately by a graphical user interface or with the help of a script resembling the programming language C. In addition the running of components can also be carried out by a configurable probabilistic model. Then the learner gets the chance of measuring running processes intensively and evaluating them graphically. The range and the complexity of experiments and measurements can be adjusted to the learner's standard of knowledge.

Communication between the components

In order to simulate a computing system it is divided up into its functional components. It is possible to distribute the single components and so the need of resources on different computers. These computers are linked via LAN or the Internet. In order make the simulation of the cooperation of several components possible, these components must exchange respective dates (e.g. for service calls and results) among each other. Here we have used the widespread protocol SOAP (Fig. 2). It supplies us with a simple and transparent mechanism of exchanging structured information. By this the computers, among which information exchange is to take place, can be located in a decentralized, distributed environment. SOAP codes request and result with the simple data protocol XML. For the exchange of data the proven protocol HTTP is used. As SOAP is independent from operating systems, programming languages and object models it is able to connect different platforms and easy to implement. Further advantages are standardization, openness, robustness and scalability. That is why it is not difficult to extend the given system by new gadgets or different interpretations.

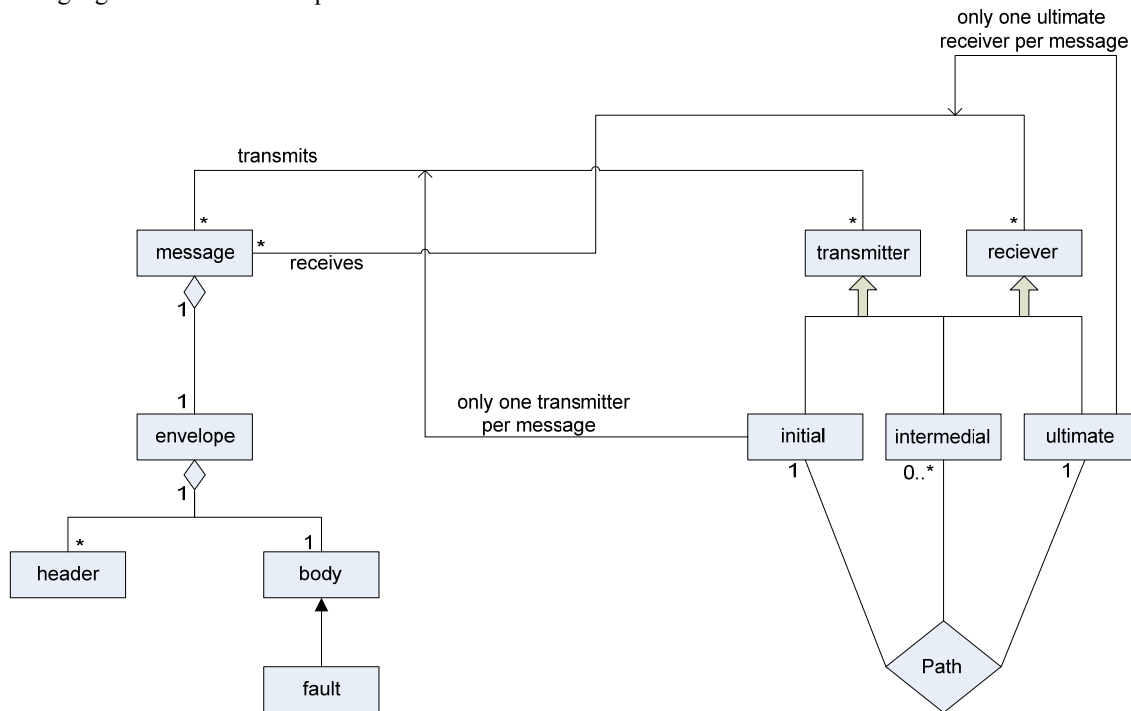


Figure 2: UML diagram of SOAP service call

Graphical user interface

The graphical user interface is a central entrance point for the usage of our framework. At the beginning of a simulation the configuration data are transferred to the single components. Then the control of the simulation can be carried out with the help of a script resembling C or by the configuration of a stochastic model.

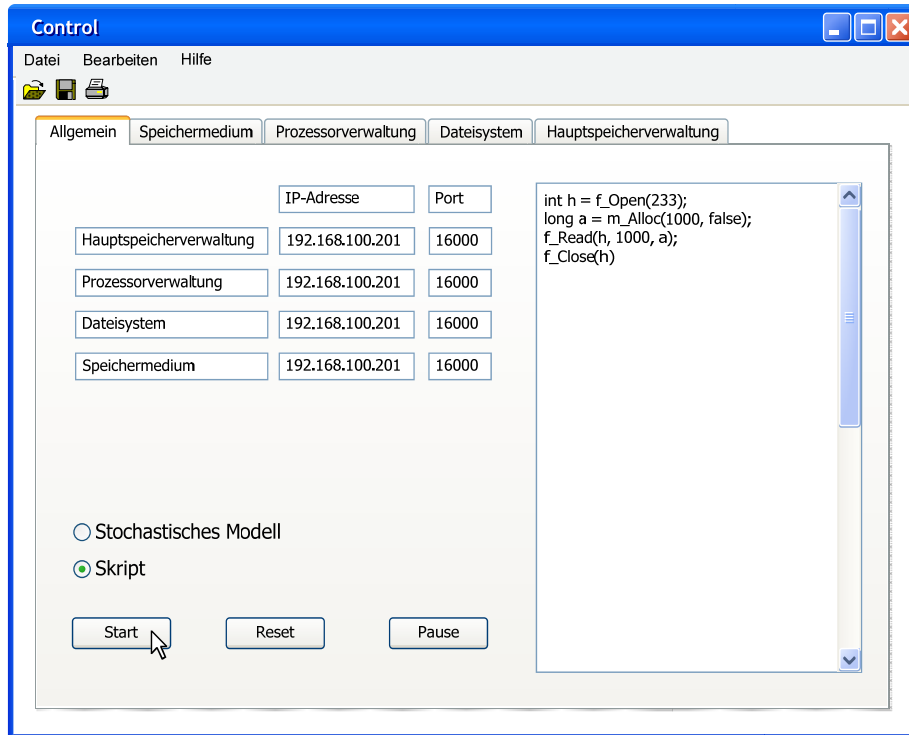


Figure 3: Graphical user interface

Components

The framework consists of the following components: storage medium, memory management, processor scheduling and file system management. Now let us have a closer look at them. In addition to these simulation components there are components for generating load and the evaluation of experiments by numbers and graphics.

Memory Management

The component “memory management” of the computer system simulation includes numerous configuration varieties. At first its size can be given. But also hardware oriented features like latency period and clock frequency can be configured. Furthermore kinds of addressing like absolute or relative storage are taken into consideration. It can be regulated if the processes running in the computer system are given virtual storage or if a swap file should be used on request.

Besides its complex configuration varieties memory control offers the following services:

- assigning memory space,
- releasing memory space,
- writing data into memory,
- reading data from the memory, and
- querying memory still available.

As a whole the component “memory management” can be applied to simulate various (memory) environments like a simple embedded system as in coffee machines and watches as well as more complex micro controller systems and the standard PC.

Storage medium

In computer systems different gadgets are used for long-term storage of data. These are shaped in the “storage medium” component in order to use them for the file system. Different features of different storage media e.g. hard disks, optical drives, tape drives and RAM-disks are simulated. It is absolutely necessary to differentiate between different gadgets, because all storage media contain different features.

E.g. a hard disk has got different average access times resulting from the arm movements of the writing-/reading head. These are not included in a RAM-drive which uses electronic storage. The total storage capacity with optical drives is standardized, e.g. with CD-R 650 megabytes or 700 megabytes can be laid down. The numerous parameters of all storage media are separately adjustable. A virtual disk is a logical resource which provides the services of

- reading a physical record, and
- writing a physical record.

So the features of different storage media are abstracted from and the functionality is standardized.

File system

Computer systems which are used for data processing always support file systems on a so-called “virtual disk”. As a rule such data systems consist of these parts: registry, allocation unit and free space management, which administrate a data carrier separated in clusters. For each of these units of the file system different algorithms can be chosen. Hereby known file systems like FAT and ext2 can be simulated but also completely independent types can be generated.

Services provided by the file system correspond to the known file operations from many programming languages:

- create file,
- open file,
- close file,
- delete file,
- read data from the file or write data into a file,
- move read/write pointer freely, and
- realize size of the file.

Processor scheduling

In modern computer systems many processes run at the same time. Then these must be distributed on a few processors. Different criteria decide which processor gets which process. This is called processor scheduling. The assigning of processors can happen to various algorithms. The component of processor scheduling comprises besides scheduling also the configuration of features of processors. There is the change of establishing one or more processors in a system. For this the principle of architecture “Symmetric Multiprocessor System” (SMP) is used. The clock frequency of each processor is adjustable.

The following services are provided:

- create process,
- terminate process,
- make system call,
- request waiting time in seconds, and
- request calculating time in clocks.

Summary

At the conference the current state of realization will be presented. The use of simulations created and finished up to now in our teachings already shows that students deepen their theoretical knowledge by dealing with the subject interactively and have better learning results.

In the future the partial exchange of single operating systems components in real systems is possible. As a matter of principle, extension by other simulating components, measurements on real machines or the use of other operating systems are possible, too. The sustainability of the project is guaranteed by its modular concept, the use of standardized protocols, and the freely accessible programming language JAVA for the central components.

References

Kalfa et al. (1997), *Praktische Übungen und Experimente in Betriebssystemen*, TU Chemnitz, Betriebssysteme, TR 3.

Koehler, F., Kroeger, R. Kalfa, W. (2002), *Integrating Simulators and Real-Life Experiments into an XML-based Teaching and Learning Platform*, edm, Denver (co).

Lucke, U. & Tavangarian, D., *Turning a Current Trend into a Valuable Instrument: Multidimensional Educational Multimedia based on XML*, ED-Media 2002.

Wissenswerkstatt Rechnersysteme, (<http://www.wwr-projekt.de>).

```

@inproceedings{FKW20050SF,
author      = {Volker Fickert and Winfried Kalfa and Thomas Weise},
title       = {Framework for Distributed Simulation and Measuring of Complex Relations
of Operating System Components},
booktitle   = {Proceedings of World Conference on Educational Multimedia, Hypermedia
and Telecommunications (EDMEDIA) 2005},
series      = {World Conference on Educational Multimedia, Hypermedia and
Telecommunications (EDMEDIA) 2005},
editor      = {Piet Kommers & Griff Richards},
pages       = {3865--3870},
publisher   = {Chesapeake, VA: ACE},
year        = {2005},
month       = jun,
affiliation = {Chemnitz University of Technology},
location    = {Montreal, Canada},
note        = {The software and more information about this work can be found here.\\
The work is online available at
#FKW20050SF. \\
The publication can be downloaded at
http://www.it-weise.de/documents/files/FKW20050SF.pdf. \\
The presentation can be downloaded at
http://www.it-weise.de/documents/files/FKW20050SF\_slides.pdf. \\
Contact Thomas Weise at tweise@gmx.de or http://www.it-weise.de/.},
copyright   = {restricted},
abstract     = {Collecting experience with the help of experiments makes it much easier
to understand complex relations. But with modern computing systems it is
hardly possible to get a direct insight into internal processes. A
current project for the simulation and realization of measuring of
processes in operating systems is presented in my lecture. Its main
point is the visualization of such processes. Because of that
distributed applications which simulate different components of an
operating system or which, with the help of appropriate drivers, are
able to carry out measurements on real machines were drawn up with JAVA.
During the simulation and measurements learners can influence parameters
of different components and experience their effects directly. Thus in
teaching operating systems it will be possible to illustrate processes
which are difficult to understand and to influence complex relations
immediately. \\
The software and more information about this work can be found here.},
contents    = {* Introduction\\
* Communication between the Components\\
* Graphical User Interface\\
* Components\\
* Summary\\
* References},
keywords    = {Operating System, Simulation, Visual Simulator, Java},
language    = {en},
url         = {#FKW20050SF}
}

```