A MULTIUSER ENVIRONMENT FOR REMOTE EXPERIMENTATION IN CONTROL EDUCATION

Christof Röhrig and Andreas Bischoff

Department of Electrical Engineering
and Information Technology
University of Hagen
D-58084 Hagen, Germany

Abstract:
This paper presents a platform independent approach to a multiuser remote experimentation system. It addresses a wide area of problems occurring in conjunction with remote laboratory experiments, starting with access management procedures via remote control to network-based analysis of measured data. Also distributed multiuser related problems like interaction and shared resources are covered. The introduced collaborative environment allows the experimentation in a team. The group is able to interact and to discuss the results of their work. A real collaboration like in local experimentation is possible. Students have access to the experiments via Internet from anywhere at any time. They control the experiments exclusively with their standard Web browser, no additional software as analyser tools are needed. The remote laboratory is based on a client/server architecture, which is mainly implemented in the Java programming language. The methods and software modules developed for the lab are generic and frequently used in several remote experiments at some German universities.
Copyright © 2001 IFAC

Keywords: Laboratory education, Teleoperation, Control education, Educational aids, Multimedia, Multiuser, Virtual reality

1. INTRODUCTION

The University of Hagen is the first and only university in German-speaking countries, which is (almost) exclusively based on distance teaching methods. As one of the largest universities in Germany it provides university-level education and related degrees. About 80% of the students are already professionals which study mainly in the evening and on weekends. The Internet becomes increasingly important as a medium for knowledge distribution and even as a learning environment.

In distance teaching, laboratory experimentation is inconvenient because the students usually have to be physically present in the universities' labs.
One solution to avoid this disadvantage is virtual experimentation. In this paradigm the experiments are simulated and visualised by means of virtual reality (Schmid, 1999). Another concept is remote control of laboratory experiments. Early implementations are reported in (Aktan *et al.*, 1996) and (Henry, 1996). Providing remotely accessible experiments, unique or expensive equipment can be shared between different universities. So, a larger number of laboratory resources is available, and students can choose from a variety of lab experiments.

In local laboratory experimentation students usually work together in groups of two or more. This learning paradigm is often called collaborative learning. Collaborative learning develops skills for solving problems in a team. The underlying premise of collaborative learning is based upon consensus building through

cooperation by group members, in contrast to competition in which individuals best other group members. Learning members of the group will usually organize their activities themselves and decide upon the roles of the different members via consultation and negotiation. (Kirscher, 1999)
With the rapid expansion and availability of communication and information technologies, collaborative learning can also be done effectively in a virtual environment at a distance. Collaborative virtual environments bring together users, which are geographically distributed, but connected via a network.

The remote laboratory at the University of Hagen is used for teaching of 'control theory'. The experiments use simple controllers with fixed structure. Students select the controller algorithm and define controller parameters remotely. There are some slightly different approaches reported, where students upload the controller algorithm to the experiment (Piguet and Gillet, 1999) or where the controller algorithm is optionally executed on the client computer (Overstreet and Tzes, 1999).

2. ANALYSIS OF REQUIREMENTS

In order to build a successful setup for remote experimentation a number of requirements have to be fulfilled:

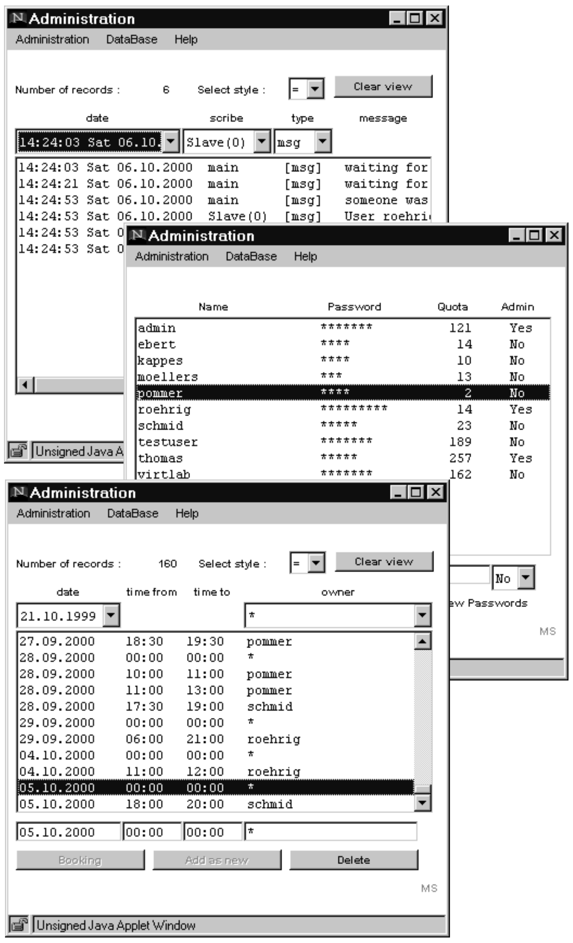
- *Telecontrol:* Computer based controller implementation is necessary for telecontrol of the experiment. In the simplest case, users change only parameters of the controller. In more advanced approaches, students upload the controller algorithm to the experiment (Piguet and Gillet, 1999) or the controller algorithm is optionally executed on the client computer (Overstreet and Tzes, 1999).
- *Telepresence:* An important aspect is to transport the feeling of a real experiment to the remote user. A video and audio broadcast can provide the remote user with the feeling of being physically present at the location of the real experiment. With the visual feedback, the user supervises the experiment and checks whether the process performs as expected.
- *Data collection:* For system identification purpose and controller evaluation, it is necessary to collect the relevant data of the process. These data have to be stored on the server for download and additional processing by the students.
- *Scheduling:* As only a group of students at a time receives access to an individual experiment, schedules and exclusive access procedures to the experiment are necessary. Users should be able to book experimentation time in advance. They should carry out the whole booking procedure by themselves in order to choose the time most appropriate to their needs.
- *Security:* The main requirement on the server side is the safety of the experiment and of the server computer. The experimental plant has to be protected against any action that can damage or destroy it. For this reason, all commands given to the plant controller must be analyzed and dangerous controller settings have to be avoided. If the controller algorithms can be defined by the user without any restrictions, system instabilities caused by the controller are hard to detect in advance.
- *Logging:* To evaluate students work, it is useful to log all communication between user and experiment.
- *Synchronous Communication:* To provide teleoperated experimentation to a group of students, synchronous communication techniques are required.
- *Collaborative Environment:* Virtual collaborative environments brings together users, which are geographically distributed, but connected via a network. This not only means that the users will be able to easily communicate, but also collaborate. The technology behind any synchronous collaboration tool is a mechanism that enables a user to send updates to other users about the interactions that are made in the shared environment. It is necessary for the participants to have the same view of the application in real time (Shirmohammadi *et al.*, 1998). A comparative summary of the different standards for collaborative environments is given in (Oliveira *et al.*, 1999).

3. ACCESS MANAGEMENT

The laboratory administrator creates and deletes accounts, he defines time quotas for each experiment. After a user has requested a particular experiment, he downloads the instructions of the experiment. Some advance preparations of the experiment are necessary, which includes modeling of the plant and subsequent controller design. When the learner has finished all preparations, it has to be submitted to the professor in charge of that particular experiment. The responsible teacher checks the prepared work and -if successful- allows the student to make an on-line appointment for the experiment and to allocate a certain time-quota.
So, in the next step access to a list of time slots is granted, which shows all dates available for experimentation. The appointment has to be made via an access management system for each experiment.

The user interface of the access management system is implemented in Java and executed in the Java runtime environment of a Web browser. It is subdivided into two parts, an administrator interface and a students interface (scheduler).

The administrator interface consists of several dialog boxes for creating and deleting accounts, for setting up individual time-quotas, for defining time slots and for analysing the logging messages. The dialog boxes for administration are shown in Fig. 1.


Administration
Administration DataBase Help
Clear view
Administration
Administration DataBase Help
ew Passwords
MS
Administration
Administration DataBase Help
Clear view
Booking
Add as new
Delete
MS
Unsigned Java Applet Window

Number of records : 6 Select style : =

date	scribe	type	message
14:24:03 Sat 06.10.	Slave(0)	msg	

```
14:24:03 Sat 06.10.2000  main      [msg]  waiting for
14:24:21 Sat 06.10.2000  main      [msg]  waiting for
14:24:53 Sat 06.10.2000  main      [msg]  someone was
14:24:53 Sat 06.10.2000  Slave(0)  [msg]  User roehri
14:24:53 Sat 0
14:24:53 Sat 0
```

Name	Password	Quota	Admin
admin	*******	121	Yes
ebert	****	14	No
kappes	****	10	No
moellers	***	13	No
pommer	****	2	No
roehrig	*********	14	Yes
schmid	*****	23	No
testuser	*******	189	No
thomas	*****	257	Yes
virtlab	*******	162	No

No

Number of records : 160 Select style : =

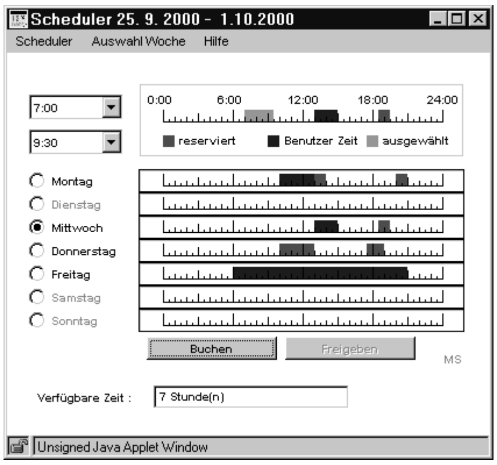
date	time from	time to	owner
21.10.1999			*
27.09.2000	18:30	19:30	pommer
28.09.2000	00:00	00:00	*
28.09.2000	10:00	11:00	pommer
28.09.2000	11:00	13:00	pommer
28.09.2000	17:30	19:00	schmid
29.09.2000	00:00	00:00	*
29.09.2000	06:00	21:00	roehrig
04.10.2000	00:00	00:00	*
04.10.2000	11:00	12:00	roehrig
05.10.2000	00:00	00:00	*
05.10.2000	18:00	20:00	schmid
05.10.2000	00:00	00:00	*

Fig. 1. Administration Dialogs

3.1 *Scheduler*

When students have received access to an experiment, they are able to book experimentation time. For logging into the server with the help of the Web browser, the student has to enter a valid user ID and password in order to pop up the scheduler dialog which is shown in Fig. 2. The user selects a week from a predefined list according to his needs.

Then the calendar shows the chosen week with free time slots indicated. The user can reserve the desired time directly with this calendar menu. Remaining times from the given time quota are monitored and available for future appointments. The actually booked appointment is stored in a database and is used for access control. At the booked time the access management system gives the user exclusive access to the experiment.



7:00
9:30
Mittwoch
Verfügbare Zeit : 7 Stunde(n)

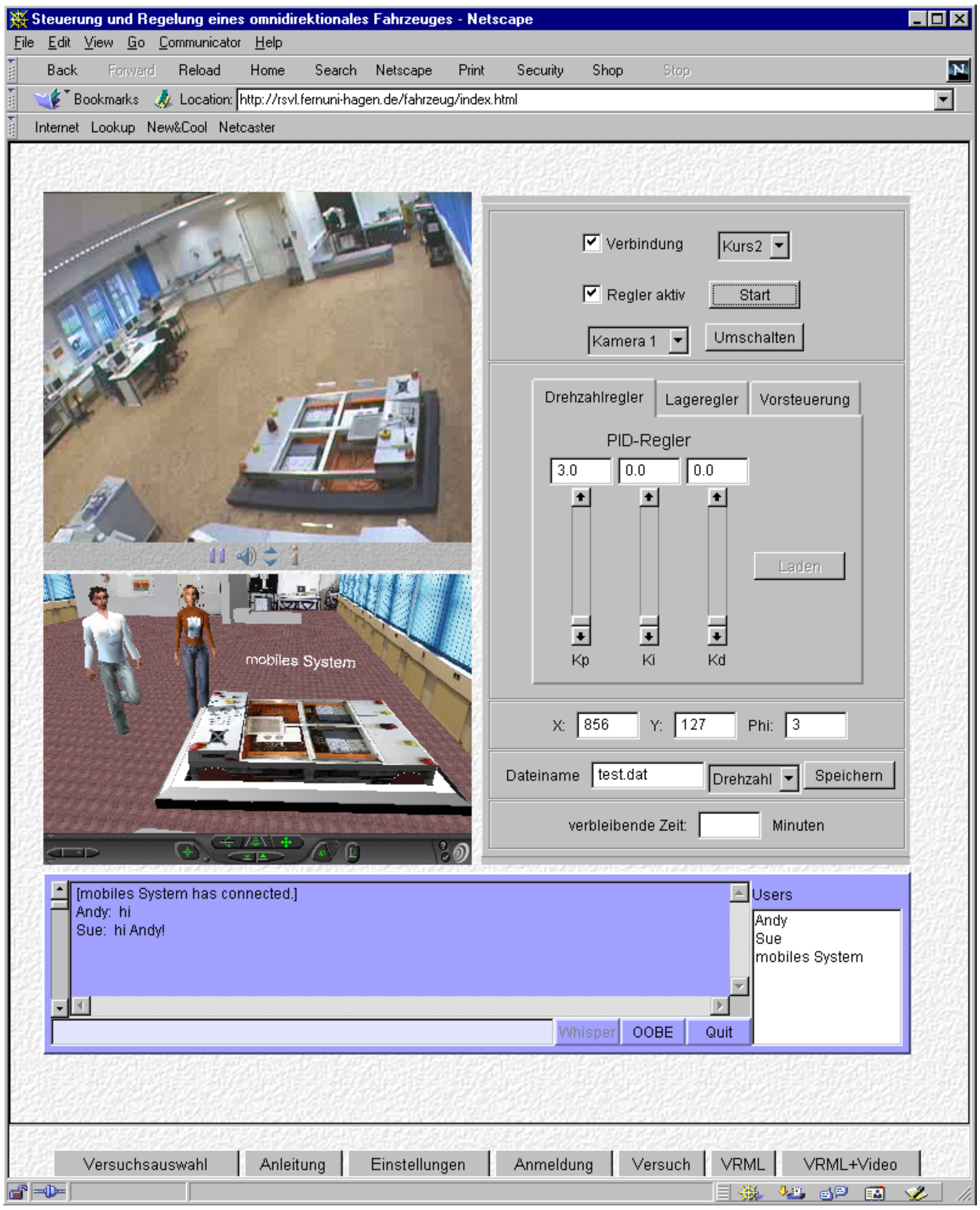
Fig. 2. Scheduler Window

3.2 *Server and Database*

The server is implemented as Java application. The communication between the server and the database is managed using the Java Database Connectivity (Sun Microsystems, 2000*a*). JDBC is an application programming interface (API) that enables access to database management systems (DBMS) from the Java programming language. It provides cross-DBMS connectivity to a wide range of Structured Query Language (SQL) databases. The SQL database mSQL (Hughes Technologies, 1999) is used with mSQL-JDBC (Center for Imaginary Environments, 1999). mSQL-JDBC is a database access API for the mSQL database engine that conforms to the Java JDBC API. The mSQL database and the mSQL-JDBC is freely available for non-commercial use. The database consist of three tables. This are the accounts table, the schedule table and the logging table. The user table stores user names, password, time quota and administrator flag. If the administrator flag is set, the user is administrator and manages the database. The logging table stores the log messages of the experiment. Messages include logging time and date, type of message and module name. Communication between client and server via Internet based on the ISO/OSI model and enhances the TCP/IP protocol. Messages between client and server are encrypted before they are sent via Internet. The encrypter uses the affine crypting algorithm with synchronised pseudorandom key generator (Straube, 1998).

4. SYNCHRONOUS COMMUNICATION TECHNIQUES

To provide teleoperated experimentation to a group of students, typical synchronous communication techniques like video-conferencing are not suitable because of bandwidth limitations. A video-conference with more than two communication partners is a typical point-to-multipoint application. If a true collaboration of all partners is desired, the partner with the


Steuerung und Regelung eines omnidirektionales Fahrzeuges - Netscape

Location: http://rsvl.fernuni-hagen.de/fahrzeug/index.html

[✓] Verbindung Kurs2

[✓] Regler aktiv

Kamera 1

PID-Regler

Kp	Ki	Kd
3.0	0.0	0.0

X: 856 Y: 127 Phi: 3

Dateiname test.dat Drehzahl

verbleibende Zeit: Minuten

```
[mobiles System has connected.]
Andy:  hi
Sue:  hi Andy!
```

Users
Andy
Sue
mobiles System

File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks
Internet Lookup New&Cool Netcaster
Start
Umschalten
Drehzahlregler Lageregler Vorsteuerung
Laden
Speichern
mobiles System
Whisper OOBE Quit
Versuchsauswahl Anleitung Einstellungen Anmeldung Versuch VRML VRML+Video

Fig. 3. Web Browser with JMF, Virtual Reality 3D-Chat and Control Applet

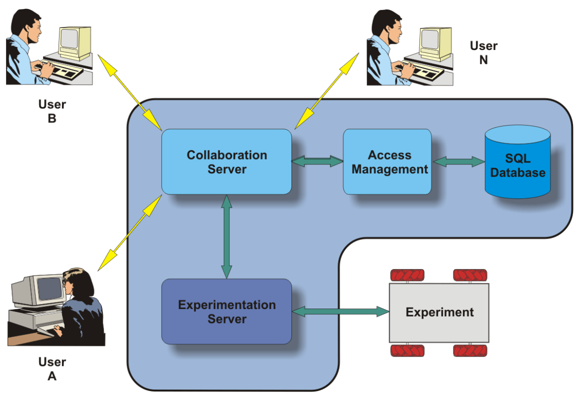


Fig. 4. Communication Structure

smallest bandwidth limits the communication. Our application of a teleoperated laboratory requires real interaction between the students and the tutor, so a bandwidth-saving way of interaction is required as an alternative to the video based communication. Pure text-based communication (Chat) does not meet our requirements because a multi-user teleoperated laboratory application needs the possibility of real interaction. The tutor has to be enabled to introduce and to explain the details of the experiment by some kind of visual representation of the experiment. In this collaborative virtual environment only one student at a time has active access to the experiment. In a 3D-chat an *avatar* tries to mimic the behavior of the user in virtual reality. The *avatar* plays the role of the video in a video-conference. Participants can see other users as in real world. 3D-chats have lesser bandwidth requirements than video-conferences, because only events are transmitted.

4.1 *Communication Structure*

The communication structure is based on a client/server architecture written mainly in the Java programming language. Students may work on any platform that supports a Web browser with a Java runtime environment. Java is used to eliminate the operating system problem of heterogeneous environments, such that users are not restricted in their choice of a resource. This is specially important for distance education since some users might choose UNIX-workstations, while others might prefer Windows 95/98/NT PCs or MACs. The introduction of Java helped to overcome these problems. The local Web browser is the only user interface to the experiment. The browser loads the client software as Java applets from the server and starts them. Due to the modular structure of the system, extensions with new features are easy to implement. The Web server provides the HTML documents, the VRML scene, VRML avatars and all Java applets. The server hardware includes a video capture card and a sound card for video and audio grabbing. The real-time controller of the experiment is usually implemented on a different computer hardware with a real-time operating system. The communication structure is generic. Therefore, it can be used for different experiments. Java applets on the client's side allow the continuous improvement of the software since the applets are loaded when they are needed. Applets are always up-to-date so no user software upgrading is necessary when the software is exchanged.

The collaborative environment is divided into two main modules: a rendering and graphics part on the client side and a communication middleware on the server side. On the client side VRML is used to display the virtual 3D environment. VRML as a text-based language is a powerful, nevertheless simple language to build 'virtual worlds', which include 3D objects, light sources and animations. VRML specifies an External Authoring Interface (EAI) which can be used by external applications to monitor and control the VRML environment. This is used to update the virtual world with data of the experiment and positions of the other users. User interfaces via comfortable Java applets can be build in order to give access to the VRML environment and to allow higher-level modifications. The communication middleware is based on the open-source VRML-Multi-User-Software VNET which realize its functionality by Java-VRML coupling via the EAI. VNET itself is a pair of client and server software implemented in Java (White and Sonstein, 1998). The server is realized as a Java application which communicates with all clients and provides them with updates of the 3D-scene. The client applet controls the local VRML-browser-plugin via the EAI to update the scene (the positions of other users) and senses the local user movements to send new positions to the server. The communication protocol between client and server is well documented, so it is possible to interface the VNET multi-user software with the teleoperated laboratory server-system. The lab server acts as an additional client of the VNET server. This additional client module provides the VNET server with the position and orientation changes during the teleoperated experiment. The remote user is able to see the virtual representation (the avatar) of the experiment (a vehicle) in her/his VRML-browser window.

5. EXPERIMENTATION INTERFACE

At the date of the experiment, the user connects to the server with a Web browser. The browser loads the Web page of the experiment. Two Java applets are included in the page. One applet receives the live video and audio stream of the laboratory and the other applet controls the experiment. Fig. 3 shows the Web browser with the video/audio and control applet. Details of the experiment itself are reported in (Jochheim and Röhrig, 1999).

5.1 *Video and Audio Stream*

A live video stream for viewing the experiment and an optional audio stream helps to provide a laboratory feeling. The Java Media Framework (Sun Microsystems, 2000*b*) was chosen to display the media streams in the browser. JMF is an API for incorporating media data types into Java applications and applets. The JMF API is a cross-platform solution written entirely in the Java programming language. Just one single applet is needed to receive the video and audio stream from the server. This applet is stored on the server eliminating the need to install it at the client. Details of the implementation are described in (Röhrig and Jochheim, 1999).

5.2 *Experiment Control*

The user interface for controlling the experiment is realized in Java. After the Web page is loaded, the applet starts and an authentication dialogue and a control window pop up. The user must enter a valid login name and password to access the server. When the login process is completed, the applet opens a TCP/IP connection to the access control module of the server.

The server reports the total time still available for the current experimentation. If no more time is left, the server cuts the students connection to the experiment.

6. DATA VISUALISATION AND ANALYSIS

For visualization of measured data, as well as support of the necessary system identification and controller design, a Java program was developed (Ondrejková, 1998). The program is executed online as applet in a browser or offline as an application in a Java Runtime Environment. The program is written in 100% pure Java. It requires a Java Runtime Environment (JRE) of 1.1.5 or higher. This JRE is included in common Web browsers (Netscape Communicator 4.08, Microsoft Internet Explorer 4).

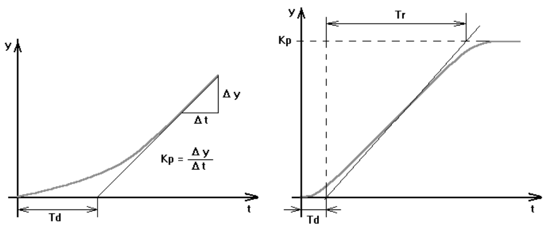


Fig. 5. Function Approximation

6.1 *System Identification*

The first part of the experiment is to model the plant and to identify the parameters of the plant with the help of a step response signal. Measured values of the step response are stored on the server. Students analyse them with the analyser applet. The analyser applet identifies the model parameters of the plant by means of function approximation. For plants with integral behaviour it calculates the values *gain* K_p and *delay time* T_d, for those with non-integral behaviour *gain* K_p, *delay time* T_d and *rise time* T_r are calculated as shown in Fig. 5.

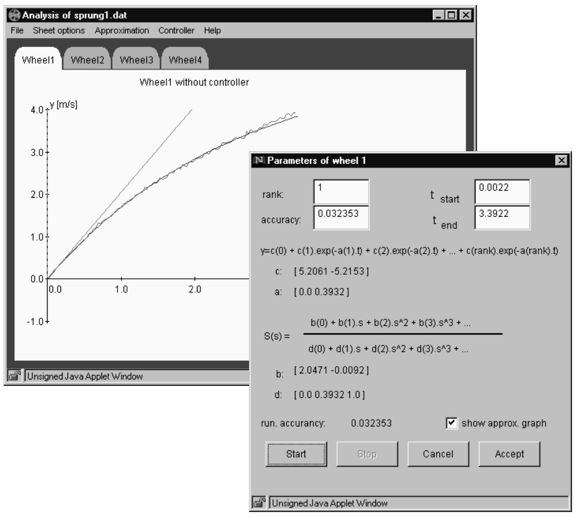
Fig. 6 shows the system identification dialog. In this dialog, the users set the rank and accuracy and start the iterative identification procedure. During the calculation, the dialog shows the actual accuracy. The approximated curve is shown after finishing calculation. The dialog displays parameters of time function and of transfer function.

6.2 *Controller Design*

Another option of the analyser tool is the support of the controller design. In the controller design dialog, users set *delay time* T_d, *rise time* T_r and *gain* of the plant K_p and select the desired design method. Actually some automatic tuning methods (Ziegler-Nichols, Chien-Hrones-Reswick,...) are implemented.

7. CONCLUSION

The contribution shows that distance education can be applied to real laboratory experiments. Even col-



```
rank:       1               t start   0.0022
accuracy:   0.032353        t end     3.3922

y=c(0) + c(1).exp(-a(1).t) + c(2).exp(-a(2).t) + ... + c(rank).exp(-a(rank).t)
   c:   [ 5.2061 -5.2153 ]
   a:   [ 0.0 0.3932 ]

             b(0) + b(1).s + b(2).s^2 + b(3).s^3 + ...
S(s) =      ------------------------------------------
             d(0) + d(1).s + d(2).s^2 + d(3).s^3 + ...
   b:   [ 2.0471 -0.0092 ]
   d:   [ 0.0 0.3932 1.0 ]

run. accurancy:     0.032353        [✓] show approx. graph
```

Fig. 6. System Identification

laboration in a team is possible. Remote laboratory users are able to interact and discuss the results of their work. A real collaboration like in local experimentation is possible. On the client side, there are only some minor requirements. Students are able to use the experiments with a Web browser and Java runtime environment. In the laboratory, the experiment must be adapted to the requirements of remote control. The server exclusively uses software that is available for free. All software developed for the lab has been implemented in the programming language Java. Therefore, it can be easily adapted to different platforms.

8. REFERENCES

Aktan, B., C.A. Bohus, L.A. Crowl and M.H. Shor (1996). Distance Learning Applied to Control Engineering Laboratories. *IEEE Transactions on Education* **39**(3), 320–326.

Center for Imaginary Environments (1999). mSQL-JDBC. http://www.imaginary.com/Java/Soul/.

Henry, J. (1996). Controls Laboratory Teaching via the World Wide Web. In: *Proceedings of the ASEE Annual Conference*. Washington, USA.

Hughes Technologies (1999). Mini SQL (mSQL). http://www.Hughes.com.au/products/msql/.

Jochheim, A. and C. Röhrig (1999). The Virtual Lab for Teleoperated Control of Real Experiments. In: *Proceedings of the 38th IEEE Conference on Decision and Control*. Vol. 1. Phoenix, USA. pp. 819–824.

Kirscher, P. (1999). Using Integrated Electronic Environments for Collaborative Teaching/Learning. In: *Proceedings of the 8th Annual Conference of the European Association for Research on Learning and Instruction*. Göteburg, Sweden.

Oliveira, J.C., S. Shirmohammadi and N.D. Georganas (1999). Distributed Virtual Environment Standards: A Performance Evaluation. In: *Proceedings of the 3th IEEE/ACM International*

Workshop on Distributed Interactive Simulation and Real Time Applications. Greenbelt, USA.

Ondrejková, R. (1998). Graphical User Interface to Evaluate Experiments in Virtual Lab. Master's thesis. University of Hagen, Slovak Technical University Bratislava.

Overstreet, J.W. and A. Tzes (1999). An Internet-Based Real-Time Control Engineering Laboratory. *IEEE Control Systems Magazine* **19**(5), 19–34.

Piguet, Y. and D. Gillet (1999). Java-based Remote Experimentation for Control Algorithms Prototyping. In: *Proceedings of the 1999 American Control Conference*. Vol. 2. San Diego, USA. pp. 1465–1469.

Röhrig, C. and A. Jochheim (1999). The Virtual Lab for Controlling Real Experiments via Internet. In: *Proceedings of the 11th IEEE International Symposium on Computer-Aided Control System Design*. Hawaii, USA. pp. 279–284.

Schmid, C. (1999). A Remote Laboratory Using Virtual Reality on the Web. *Simulation* **73**(1), 13–21.

Shirmohammadi, S., J.C. Oliveira and N.D. Georganas (1998). Applet-Based Telecollaboration: A Network-centric Approach. *IEEE Multimedia* **5**(2), 64–73.

Straube, M. (1998). Skeleton of Internet Application out of Deference Safeness. Master's thesis. University of Hagen, Slovak Technical University Bratislava.

Sun Microsystems (2000*a*). Java Database Connectivity. http://www.java.sun.com/products/jdbc/.

Sun Microsystems (2000*b*). Java Media Framework. http://www.java.sun.com/products/java-media/jmf/.

White, S. and J. Sonstein (1998). VNet. http://www.csclub.uwaterloo.ca/˜sfwhite/vnet/.