Organising Tele-experiments for Control Education

Pavol Bisták and Katarína Žáková

Abstract -- **This paper presents experience in building remote control laboratories. It shows various possibilities of organizing real experiments via Internet that foster students to an active and creative learning by doing.**
The different approaches how to extend the existing laboratory infrastructure are presented. The paper includes detailed description of a server-client application that has to be implemented. Except of this safety conditions and necessary hardware changes are discussed.

Index terms -- **tele-experiment, remote control, virtual laboratory, Matlab, client-server applications**

I. Introduction

The rapid development in the information and communication technologies enables to increase the diversity of means used in teaching and education. Internet technology offers more ways how to reach students, transfer knowledge and develop skills. For the branches of education that require the development of hands-on abilities with laboratory equipment – namely in engineering – the possibility to tele-operate equipment has opened new horizons for tele-education. Virtual laboratories aim at providing via the Internet the hardware components and associated practical skills that complement WEB-based course material [3].
The necessity of learner's experimental work in laboratories represents one important aspect of engineering education. Students solve practical problems and gain experience and practice needed for their future career. Introduction of real systems increased transparency of the solved examples.
The main motivation for using of plants in the educational process is clear physical "visibility" of the controlled dynamics, and also the necessity to exercise all design steps starting with the plant identification and ending with the evaluation of the control results achieved with the particular model. In fact, the model by itself can be presented as a virtual device whose dynamics is described by differential equations or as a real plant. With the development of new computer technologies, an interactive multimedia programming language JAVA, and the WorldWideWeb, it is now possible to simulate the virtual model on a computer and with Internet access to offer it as an animation to students in frame of "virtual laboratory" via the WWW or CD-ROM. However, using the animation models cannot substitute the work with real physical plants that always demonstrate some unmodelled dynamics, parasitic noise, friction, etc. Unfortunately, the number of students is high in comparison with the number of available real plants. Building remote laboratories that gives learners access to laboratories via Internet represents a possible solution of this problem (Fig.1).

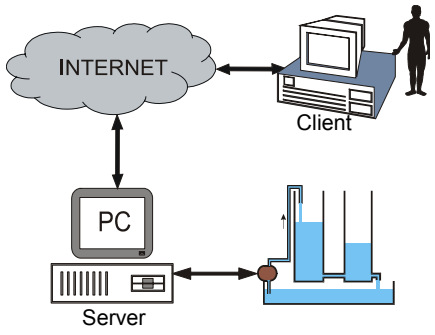

Fig. 1. Experiment control with the real plant through Internet

Expensive and elaborate experiments can be set up once and can be made available as so-called tele-experiments all over the country or even the whole world via Internet. Experiment-oriented problems can be offered without the overhead incurred when maintaining a full laboratory.
At the preparation of tele-experiment, it is important to take into account that students should be actively involved in it. They must be able to change parameters and to collect measurement data. As far as possible, the running experiment should be observable through Internet.
Finally the time for conducting an experiment must be short enough to avoid longer waiting queues.

Pavol Bistak and Katarina Zakova are with Department of Automation and Control, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, Ilkovicova 3, 812 19 Bratislava, Slovakia. Email: {Pavol.Bistak, Katarina.Zakova}@elf.stuba.sk.

II. Concept of the System

The concept is based on the existence of classical control engineering laboratories with experiments carried out in the presence form. The already built infrastructure is going to be exploited. In education of control engineers we usually use Matlab software for implementing various control strategies. Due to different toolboxes (like Real Time Control, Windows Target and xPC Target) Matlab with Simulink is not only simulation tool but nowadays it enables to control real plants. There exist many experiments using this platform. Our aim was to open laboratory equipment to more students, especially students who can not be present. This extension requires only small changes in laboratory hardware equipment and a little bit more programming work in software equipment [2].

To allow remote users to run experiments we have chosen the Internet connection, as this type of connection is widely accessible. Then it has been necessary to program a client-server application that handles the data flow between the real plant placed in the laboratory and the remote user. The client-server application is Web-based application so the remote user can reach experiments within the familiar environment of his/her Web browser. The client-server application communicates through TCP/IP protocol.

On the server side the application has to communicate with the Matlab application that directly controls the real plant. There can be different type of communication but it is advantage when the server part of the application resides on the same PC that controls the real plant. Then within one operating system the data are exchanged more easily.

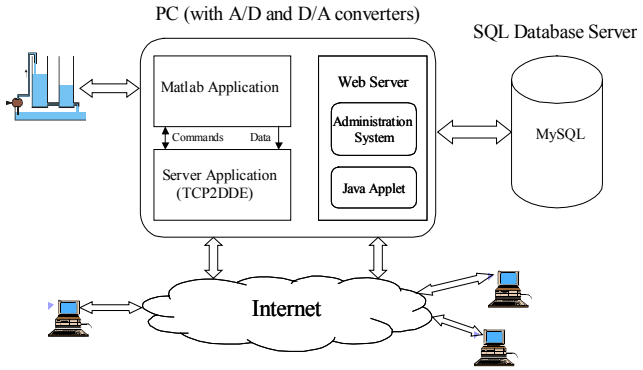

Fig. 2. Overview of the concept

The transfer channel is complicated and moreover the Internet connection is not very reliable one. Therefore it is advisable to divide the whole program application for tele-experiments into following parts

- Local control (verified control strategies implemented locally with the use of the Matlab application)
- Server application (this serves for handling data flow between the remote client and the real plant, it transfers user's commands and transmits responses)
- Client application (remote user interface that enables to control a specific real plant usually in the form of JAVA applet running within Web browser)
- Administration (the application that covers administrative issues in the case of many users for several real plants)

III. Local Control

Prevailing structure of a Real Plant Control laboratory for education of control theory is characterized by pairs: a real plant and a computer equipped with a data acquisition card. A simple PC with Matlab installation offers the comfortable way how to test different control strategies. The results can be simulated but also measured from the real plant. Matlab provides a support for some A/D and D/A cards. For other cards it is necessary to write drivers.

In our Real Plant Control Laboratory one can find following physical models [4]: two tank system, magnetic levitation, inverted pendulum/gantry crane, helicopter and mining lift. All of them we are able to control within the Matlab environment. As the data acquisition cards the AD512 and MF604 were used. Each real plant has a corresponding electronic device that amplifies the signals from a computer.

Except of the mining lift all models have been controlled via Internet using tele-experiment approach. For tele-experiment purposes some of the plants have to be modified. For instance the two-tank system was equipped with an additional set of valves that enable its change from two-tank system to one-tank system and vice-versa. Also the better identification can be carried out. The helicopter model and inverted pendulum/gantry crane have to be equipped with additional sensors for reference position as they use incremental sensors.

In fact, each real plant is a nonlinear system and offers a wide spectrum of tasks. Beside identification we have implemented various linear controllers (LQ control based on linearization around fixed operating point) and nonlinear controllers (generalised exact linearization, constrained control, anti-windup PID control) or fuzzy control with neural control. Introducing tele-experiments our results can be compared with other remote users testing their algorithms on the same hardware.

IV. Server Application

The server is an application that listens at a particular port and responds to clients. It could be any standard servers like http, ftp or a custom server made for your specific application. The server application is the central part of the whole system. The features of this application in many ways define the possibilities of clients that are going to exploit its services. The server's main function in our case is to transfer commands and send the output data to a client. Another feature is that the server should co-operate with the Matlab application that really controls the plant. This co-operation is based on the data exchange. The data exchange

can use different technologies given by the actual operating system. On the Windows platform there exist several possibilities how different applications can exchange the data. These are represented by DDE (Dynamic Data Exchange), ActiveX and CORBA. In the following part there will be more detailed described two different server applications that use DDE and ActiveX components.

A. Server based on DDE (TCP2DDE)

For data exchange between two applications this server application uses the DDE communication tools that are implemented in the Windows operating system. Another standard tool is used for communication with clients – TCP/IP connection. This was the reason why the server application got the name TCP2DDE.

TCP2DDE consists of three main components as depicted in Fig. 3. These are TCP Server, DDE Client and DDE Server.

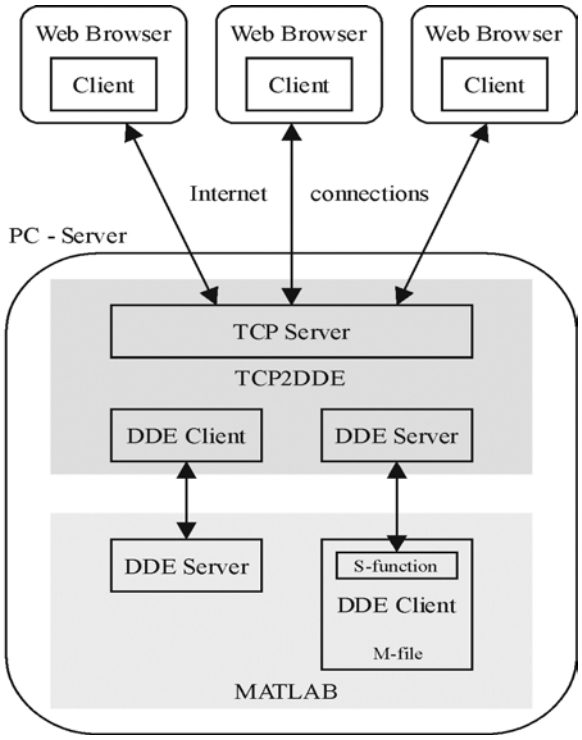

Fig. 3. Data exchange using the TCP2DDE application

After activating the TCP2DDE application it runs the Matlab application (if it is not already running). This is accomplished by the Windows function that runs the Matlab when the connection with Matlab DDE Server is required. Thus it is assured that when the Matlab is occasionally cancelled it will run again automatically.

TCP Server handles the communication with clients. Except a few specific commands it transfers all client's commands to Matlab to be executed. The transfer of commands is realized by the DDE Client component. After being activated the DDE Client establishes the connection with Matlab DDE Server. Using the DDE Items of the Matlab DDE Server (EngStringResult, EngFigureResult) the DDE Client gets Matlab answers. Then these answers are transmitted to the TCP Server that finally sends the answers to the client.

The data created during the running of experiment are delivered by a little bit different way. There is an S-function incorporated in the Simulink model of the control circuit. The Matlab DDE Client establishes the connection with the DDE Server that is always running within the TCP2DDE application. Then the data are sent by the S-function to the DDE Server, more precisely using its DDE Items. Further the data are transmitted to the TCP Server and sent to the client.

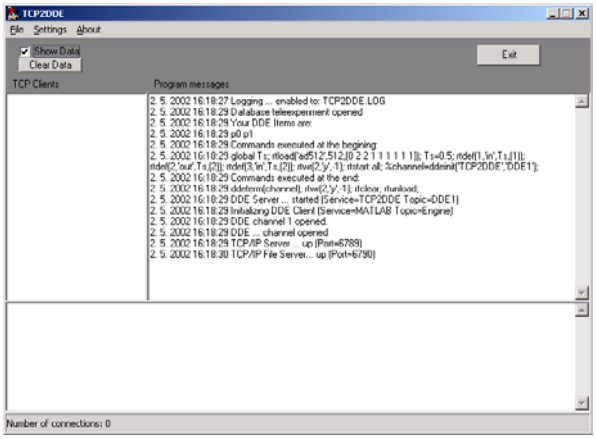

Fig. 4. The TCP2DDE application window

The TCP2DDE application has to be set up after installation. When running the application the main window appears as it is shown in Fig. 4. The Settings entry from the main menu opens the ConfigFrom dialog window that consists of several tabs. The DDE tab enables to set up the parameters of the DDE communication. In the case Matlab is the application for the DDE communication (that is our case) the parameters do not need to be changed. The variables tab sets up the variables of the DDE communication (DDE Items).

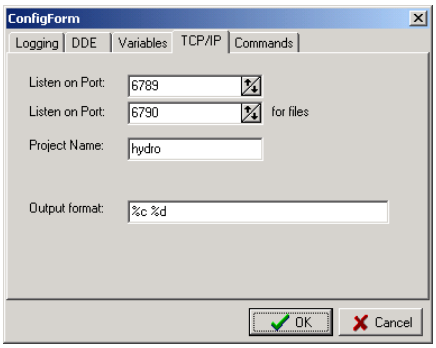

Fig. 5. TCP/IP settings

TCP2DDE is expecting the communication will be held on specific ports which numbers can be chosen on the TCP/IP tab as shown in Fig. 5. Further it is possible to change the

name of the project that represents the specific real plant. For the administrative purposes this name must agree the name stored in the database of the administrative module because it is checked when establishing connection. The Output format field gives the information about the format of the data that will be sent to a client.

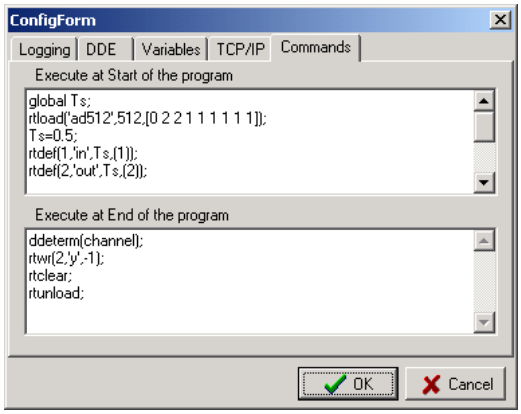

Fig. 6. Commands executed at the start and the end of TCP2DDE application

TCP2DDE solves also problems with running and stopping the Matlab simulation. On the Commands tab of the ConfigForm dialog window (Fig. 6) it is possible to set up commands that have to be executed before the control experiment starts and after it finishes. When, for the control purposes the Real-Time Workshop is used it is necessary to write corresponding S-function in the C programming language.

The TCP2DDE server application also includes the administration module that takes care about the use of one hardware by more users. The administration part will be described below.

B. Server based on ActiveX

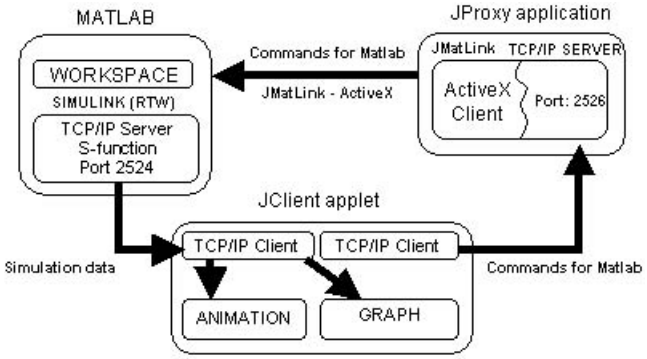

Fig. 7. Realization of the connection Java client – Java proxy server - Matlab

The another possibility of the server application is to create a Java proxy server that will interpret user commands from the client application to Matlab. For this purpose it is possible to use a dll library JmatLink [1] that accesses services and data of Matlab and enables users to connect to Matlab from Java applications. The library is implemented to the simple class JmatLink.class that can be used inside of created Java proxy server.

Proxy server was realized as a Java application with the structure shown in Fig. 7. Its input side is formed by a TCP/IP server that receives commands from the client (Java applet). These commands are transformed to the Matlab commands and sent to the Matlab workspace using methods of JMatLink library (see Fig. 8).

Fig. 8. Realization of the data transfer to Matlab

The proxy server has 3 basic functions:

- to start the real time simulation,
- to stop the real time simulation and
- to change the model parameters during the simulation running.

Its realisation is shown in Fig. 9. Before running the tele-experiment, the administrator has to set the path to the simulation model and to the file with the initial simulation parameters.

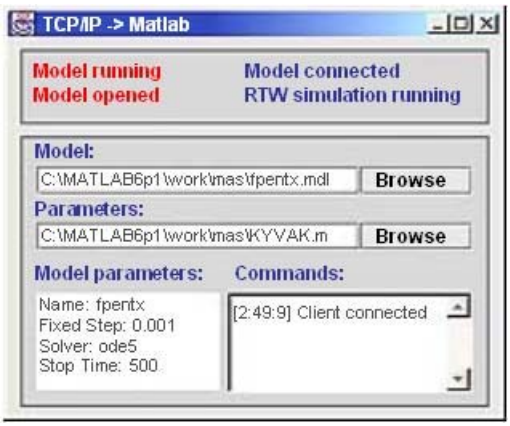

Fig. 9. Java proxy server

V. CLIENT APPLICATION

In general, TCP/IP client is an application that connects to a specific port on a TCP-IP server and exchanges data either as a stream or text.

It is installed on a web server that can be approached by a student via Internet. It is usually realized as Java applet that enables to modify controller or simulation parameters and visualize simulation results. In Fig. 10 the applet for an inverted pendulum control is shown. After its opening in an user web browser, TCP/IP client connects to the proxy server and to the S-function in the Simulink model (Fig. 7).

An user is informed about the time of connection and the connection status through short information note in the first part of the applet window („client connected“). In this moment, everything for real-time simulation is prepared.

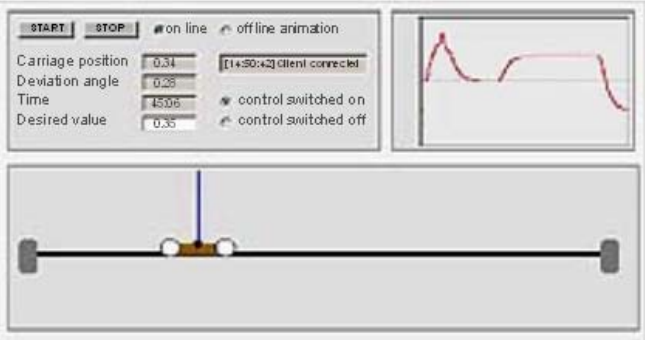

Fig. 10. Java client for inverted pendulum experiment

The user can enter a required position of the pendulum. The buttons „Start“ and „Stop“ serve for the control of simulation running. During the simulation user can follow numerical values of the carriage position, deviation angle of the pendulum, time of simulation and the graphical dependence of the carriage position. The simulation results can also be visualized through model animation that can run in two modes: on-line and off-line mode.

In the on-line mode the animation starts immediately after the beginning of simulation and in each moment user can see actual state of the pendulum in the animated model. The time between the real time experiment and the animation is synchronized. However, the velocity of this animation depends on the velocity of data transfer via Internet. If the transfer is not sufficiently fast, the animation is not capable to visualize the samples from the experiment on line because of big delays. Then, the off-line animation can be used. In that case the real time simulation is realized only numerically without the contemporaneous animation. After the simulation ends, measured data are sent to the client and the animation can be accomplished.

So, it means the Java client has to enable to user

- to control the process and
- to collect and visualize the data from the process

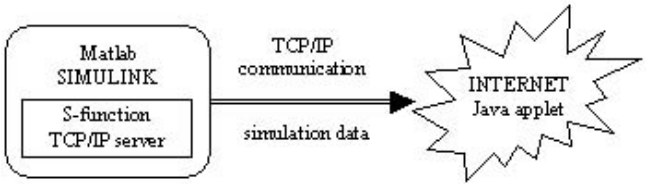

Fig. 11. Realization of the data transfer from Matlab

Connection between Matlab and Java client can be realized directly using TCP/IP communication (Fig.7, Fig.11). TCP/IP server can be realized by Matlab S-function. It enables transfer of data from Matlab to the connected TCP/IP client (Java applet). The Matlab S-function operating as TCP/IP server is placed in Matlab/Simulink model whereby the input of this block is formed by signals that should be transferred to the client.

There is another example of the client application shown in Fig. 12. This client serves for control of the two-tank system. The remote user can receive the measured data numerically and at the same time to see them in the form of graphs or simple animation.

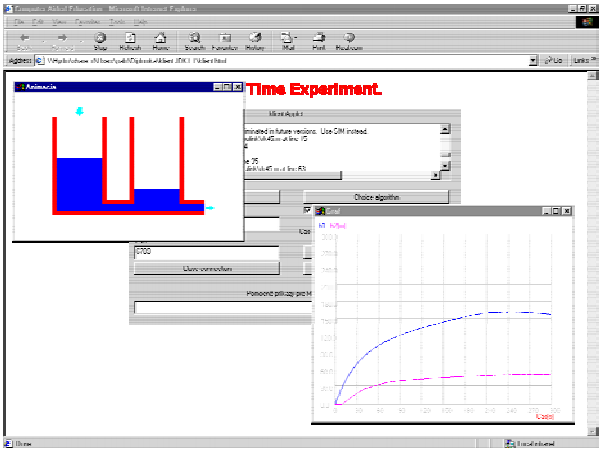

Fig. 12. Java client for the two-tank system

Similar client applications have been created for the helicopter model and the magnetic levitation system.

VI. ADMINISTRATION

Since the Java client is placed on the public accessible web server, tele-experiment could be available not only to students but also to the public community via Internet. Therefore it is necessary to ensure sharing of tele-experiment by several users and synchronization of the access (Fig. 13). It means the administration of the experiment has to be done. The main aim is to guarantee, that in one time instant only one user can work on the real equipment. For this purpose the script language PHP can be used. One of its properties is that it allows to access relational databases (SQL databases). The administration of tele-experiment should ensure the access to the experiment only for defined group of users whose access rights and time scheduling are specified in the SQL server database. After entering the system by means of login name and password and correct authentication of the user, the user is redirected to the web pages with the client applet that enables the real time experiment.

Administrator of the system has rights to create new accounts for users and to cancel old accounts. He can specify time intervals when the user can access the real experiments and accomplish the simulation. In this way the work of several users can be coordinated.

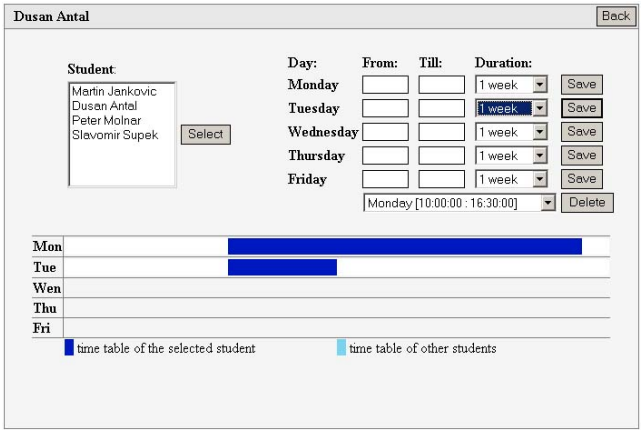

Fig. 13. Time table allocation

VII. SAFETY CONDITIONS AND RECOMMENDATIONS

In general, basic requirements for the remote laboratory operation should be fulfilled. It is necessary to ensure

- remote control of the experiment and the possibility to switch on/off device from far-out user
- stable beginning states
- security for personal and laboratory devices

The security should be guaranteed through

- hardware check (protect device or person)
- software check
- network security that includes
  1. protection against unauthorized access
  2. time limited (scheduled) access - only one person can work on device
  3. user access transparency

After verifying these it can be said obligatory conditions it is good to provide the tele-experiment with next additional features such as

- presentation of the experiment output to remote user
- compatibility with Matlab/Simulink simulation and experimental interface
- experiments administrator
- wizard (interactive study guide)

Except of this it would be very convenient to ensure also a visualization of the system behavior by the output data (graph, picture), or even by an additional video and sound transfer.

VIII. CONCLUSIONS

Our attention was dedicated to the development of tele-experiments. For local control purposes the Matlab application with its real time toolboxes have been used. The different plant models have been tested for the remote control and are now being made accessible to students and colleagues via Internet. By elements of playing introduction of the tele-experiments in the control education reasonably increases the student motivation and also develops various other skills in the signal measurement and processing.

IX. ACKNOWLEDGEMENT

The work has been partially supported by the Slovak Scientific Grant Agency, Grant No. 1/0145/03 and Grant No. KG-57-HUBA-Sk1.

X. REFERENCES

[1] JMatLink Version 1.00; http://www.held-mueller.de/JMatLink

[2] Müller S., Waller H.: Efficient Integration Of Real-Time Hardware And Web Based Services Into MATLAB, *11th European Simulation Symposium*, October 1999, Erlangen, Germany.

[3] Schmid Chr.: Virtual Laboratory for Engineering Education, *ICDE conf. Vienna*, Austria, 1999.

[4] Zakova, K., Huba, M., Zemanek, V., Kabat, M. : Experiments in Control Education, IFAC ACE 2000., Gold Coast, Australia.