Remote Laboratory Allowing Full-Range Student-Designed Control Algorithm

Laurentiu Frangu, Claudiu Chiculita

University of Galati, ROMANIA

ABSTRACT

The paper deals with the structure and the properties of a system implementing the remote laboratory, in the technical area, for university purposes. This system was intended for remote experiments on physical objects, but it allows virtual experimentation, too. The main contribution is the original software written for this laboratory, created by the authors and tested in the university. The key feature of the laboratory is that it allows the remote student to take full control of the equipment, in order to solve the didactical subject, protecting in the same time the laboratory against possible attacks. The described application is in the field of automatic control, but remote experimentation in the field of electronics or other technical subjects is also possible. The student is allowed to collect data from the equipment, to reconfigure on-line the structure of the equipment and to write his own automatic control algorithm, without being limited to predefined algorithms.

1. INTRODUCTION

The remote education is an important issue of our days, equally in academic or industrial context. The remote laboratory is part of this approach, presenting more advantages: it saves equipment, personnel for the institution and it saves time and money for the remote students. It also allows the same equipment to be used in research purposes by more teams, through Internet, and it is a good case study for the informatics and control aspects of the telepresence and teleoperation applications [1]. There are three different aspects of the remote laboratory implementation: the technical equipment used in the experiment (specific to the taught subject), the communication software and the teaching software. This paper deals with the possibility of using the equipment in the laboratory by the remote students, mainly from the point of view of the informatic system. It concerns the communication software and the solutions to some important didactical requirements.

This kind of laboratory has to observe the general requirements of such activity, presented in [2]. There are technical, didactical, financial and security issues, like the ability of protecting itself against malicious software and incorrect operation. The software described in this paper meets most of these requirements. It was validated through an application in the field of automatic control education, but it is also applicable and easy to set up in different other fields. Its key feature is that it allows the remote student to take full control of the equipment, in order to solve the didactical subject, protecting in the same time the laboratory against possible attacks. The student is allowed to collect data from the equipment, to reconfigure on-line the structure of the equipment and to write his own automatic control algorithm, without being limited to predefined algorithms.

The paper continues, in Chapter 2, with the structure and the features of the remote laboratory. Chapter 3 contains the description of the software implementation and communication solutions. The experimental results are described in Chapter 4, whereas Chapter 5 presents the conclusions.

2. STRUCTURE OF THE REMOTE LABORATORY

There are today many universities that organize remote education, including for the technical field. The courses and the simulation laboratories are distributed on electronic supports or by e-mail. However, the laboratory experiments that use physical equipment are not replaceable, because the student must have a direct contact with the real technical environment. Part of the work has to be organized in the institution, while the other part may be performed from the remote site. The usefulness of the remote laboratory proposed in this paper is justified by the opportunity of organizing the remote experiments, saving time and money for the students.

In the sequel, the structure of such a laboratory will be described. The following terms are used throughout the paper: the server is the computer situated in the laboratory, interfaced with the physical equipment; the client is the student's computer, remotely situated. A very general laboratory structure must include:

- the equipment (the regulated object, in the case of automatic control laboratory) associated with a server computer and a data acquisition card (in the case of virtual experimentation, the equipment is replaced by a simulation software);
- an interface program, running on the server computer, that drives the signals to and from the data acquisition card and manages the information about the students' work;

• a server software that deals with the communication between the laboratory and the remote students;

• the informatic environment (usually Intranet or Internet);

• a client software running on the remote (student's) computer, that provides the Graphic User Interface and communicates with the server.

The nature of the equipment depends on the subject of the laboratory, generally thought for the technical area. In the case of the control laboratory, we chose as regulated object, a single input - single output level control system, presenting nonlinear behavior. The acquisition card is simple (PCL711, Advantech), because it needs only an analog input and an analog output. A single computer runs the interface to the acquisition card and the web server, under Windows environment. Because the process is slow, the real-time capability is provided by the usual functions, avoiding the necessity of sophisticated real-time kernels.

The actions performed by the remote student with respect to the equipment are: sending batch commands and recording measured data from the regulated object, reconfiguring (even on-line) the equipment and sending an own algorithm to control the regulated object. The recorded data is useful for identification purposes. The reconfiguration means to change the hardware structure, to change setpoints or to change parameters. This is a very useful feature in the case of electronics and automatic control laboratories, allowing to change the structure of the circuits, gain, bandwidth, etc. The key feature of the laboratory is that the student is not constrained to use predefined controllers, having the possibility to write his own full-range control algorithm. In fact, he can test any algorithm he can imagine (fuzzy, neural, etc.).

Even if the mentioned application field is the automatic control, other technical areas can also benefit from the same structure. For instance, robotics, mechanics, ship model testing, data acquisition and management laboratories can use it.

3. IMPLEMENTATION OF THE SOFTWARE

3.1. Comparison to commercial products

There are not many commercial software products, allowing remote operation of the equipment, for teaching purposes. All of them are derived from simulation software. Studying the most spread simulation software products (Matlab + Simulink, LabView and Quanser) some conclusions may be drawn ([2], [3]). The available commercial products are not intended to solve the problems of the remote technical education, so they do not present facilities for the didactical aspects of the remote laboratory experiments (such as private student directory, access of the teacher, etc.). Regarding the technical aspects, in the field of automatic control, where the student must implement his own control algorithm, only the most suitable product (Quanser), poses security problems because of the use of compiled files and does not allow the change of parameters, during the experiment.

3.2. Implementation of the software

Our goal is to offer to the student the possibility to carry on laboratory experiments on the existing physical equipment. In this purpose, the client computer runs a web browser only, which opens a web page loaded from the server. The software implemented by the authors runs on the server computer and cumulates the functions:

- interface to the physical equipment (through the data acquisition card);
- interpreter for the student-written program;
- web server, dealing with all communications between server and client;
- manager of didactical information.

Figure 2 illustrates the interaction between the parts of the software running on the server and on the client. The main features of the software and the reasons for choosing this structure are described in the sequel.

Server-side executed code

There are two aspects concerning the execution: where the controller runs (or other software part that drives the physical equipment) and where the source code is compiled (or interpreted). Running the software on the client implies that the control loop includes the server, the client and the network. This approach introduces large and unpredictable dead-time in the loop, making it suitable for slow processes only [4]. In order to obtain a convenient solution also for faster processes, the presented laboratory system executes the code on the server, as presented in figure 1. The second issue concerns the type of data sent from the client to the server. As previously mentioned, sending binary code from the client to be executed on the server (this is the solution adopted by Quanser's product) creates security problems and requires a compiler to be present on the client. The adopted solution is to edit the source code of the controller at the client and to send it to be interpreted and executed at the server. To avoid security problems caused by the source code, this one is filtered by a restrictive environment prior to reach the interpreter.

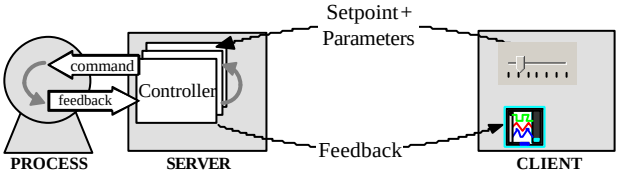

Figure 1. Server-side hosted controller

The communication solution

The communication between server and client may be done in several ways. The direct TCP connection requires

dedicated programs, running on the two computers but the resulted client is not portable. A better solution, the connection by Java applet, requires all software written in Java and a browser supporting this language. Finally, a dedicated web server, communicating with the client web general browser, is more flexible. Even if it requires more effort to write it, we chose this last solution, because of the interface with the process that had to be integrated in the server. Additionally, leaving only a web browser to run on the client provides a portable and low cost solution for the student.

Web browser client interface

The student's graphic interface is presented on the right side of Figure 2. It is loaded from the server, when starting the editing session, and it runs inside the browser. The top left frame contains the edited program, which will be sent to and interpreted by the server. The bottom left frame contains the values of the process parameters which can be changed during the experiment. The right side frames contain data received from the process and will be detailed in chapter 4.

Powerful language: Python

We chose to use Python as the student programming language. Python is an interpreted, high level, object oriented language [5]. The main reason for using it is its feature to allow building a restrictive environment that filters the interpretation of the Python client source code. This feature solves some of the security problems: the client has restricted access to the internal resources of the server computer. In the present the restrictions introduced by the authors (using the above mentioned feature) concern the access to the controlled object and to the file system.

Python is also very simple to learn; this feature has an obvious didactical motivation, because it allows shortening the student's requested time for accommodation to the language.

Other advantage of Python is that it produces short source code; as a result the code written by the student is more visible in the limited size window, provided by the web browser. A last reason for choosing this language is the fact that it is free.

Solutions for the didactical aspects

Besides the full control of the equipment, the student takes advantage of some other important features. The opening page on the server is available for any client; it contains the structure of the site, the description of the experiments, the necessary hard and soft documentations. Concerning the own work of the student, the presented software provides him with a private directory on the server, which contains his programs and data. The access is based on name and password. Only the professor has access to all directories in order to evaluate and to mark student's activity. His access is also password based.

While multiple clients have simultaneous access to the common pages of the server, the access to the equipment is available for a single client. This is why one important feature is the possibility of the student to make a reservation of the time interval, necessary for him to experiment with the equipment. This feature corresponds to the concept "unsynchronized", specific to the remote education.

4. DESCRIPTION OF THE EXPERIMENTAL SESSION

For illustrating the use of the proposed remote laboratory system, a short description of the experimental session is presented in the sequel.

First of all the student starts a web browser and loads the web page of the laboratory. There he can study online the theoretical aspects regarding the experiment he is going to take, the tasks required by the teacher and some examples of programs written in the specific language. In order to begin a session, the student must log on (name and password). Once logged in he can access his private files and can make a reservation in the laboratory schedule. If the current time corresponds to a previous reservation done by the student, he can move on and proceed to the main experiment window. To start a solution, the student must edit the source of his program and to send it to the server. The server runs this program (the controller) in the restricted environment, allowing limited access to resources, like acquisition card and file system, and denying access to sensitive ones.

During the execution, the student can read the data from the process and change the parameters. This is an important feature of the laboratory system, because these parameters can have any meaning chosen by the user. For instance, they can be used as setpoint, controller parameters, configuration parameters of the equipment, etc.

The right bottom frame contains the diagrams of process data, but it can also contain image from a video camera, showing the equipment; the upper right frame displays the output data of the program. At the end all data can be saved, both on server or client. Furthermore, the student can study how appropriate is his controller, that drives the physical equipment and can improve his solution to the problem. In the case of long-run experiments the student can interrupt his internet connection, let the process run and come later to resume his experiment or simply to collect the data. Finally the teacher reads the results and marks the activity of the students. Because the access of the remote students is not synchronous, this solution saves equipment and space in the laboratory. Usually the equipment does not need current operation and maintenance, so there is no need for supplementary personnel (the laboratory may be accessed even during the night).

The solution for the described system was adopted after testing and comparing more possible variants (such as running the code at the client). Then it was tested

inside the university (Intranet), in order to verify the didactical, technical and security functionality. During the tests the students learned quickly the language and performed the tasks asked by the teacher. So it is confirmed that the structure is functional and the goal of the work was reached.

5. CONCLUSIONS

The main features of the proposed system, which is not a commercial product, are:

- it satisfies some didactical requirements, such as easy to learn language, private directory and reservation in the laboratory schedule;
- it is a low-cost solution for the student (this is an important issue for the success of the remote learning, in virtual universities);
- it introduces laboratory experiments on physical equipments (this is not a common feature and it is important for technical education);
- it simplifies the programming and security problems, because the client runs a web browser only;
- it is a safe solution because the student has access to the allocated resources only.

Further work will concern the graphic interface, the didactical aspects and the application to other equipments.

6. REFERENCES

[1] D. Gillet, Ch. Salzmann, R. Longchamp, D. Bonvin, "Telepresence: An Opportunity to Develop Practical Experimentation in Automatic Control Education", European Control Conference, Brussels, Belgium, 1997

[2] C. Chiculita, L. Frangu, "A Web Based Remote Control Laboratory", to be presented at the conference SCI2002, July 15-th 2002, Orlando, Florida, USA

[3] D. Gillet, Ch. Salzmann, and E. Gorrochategui, "Remote Manipulation with LabVIEW for Educational Purposes", Computer Based Measurement and Automation in Education, 1998 Conference Proceedings, National Instruments, Austin, TX.

[4] M. Buss, G. Schmidt, "Control problems in multi-modal telepresence systems" in Advances in control, Highlights of ECC'99 (Paul M.Frank, Ed.), Springer 1999.

[5] Guido van Rossum, Python Documentation, www.python.org

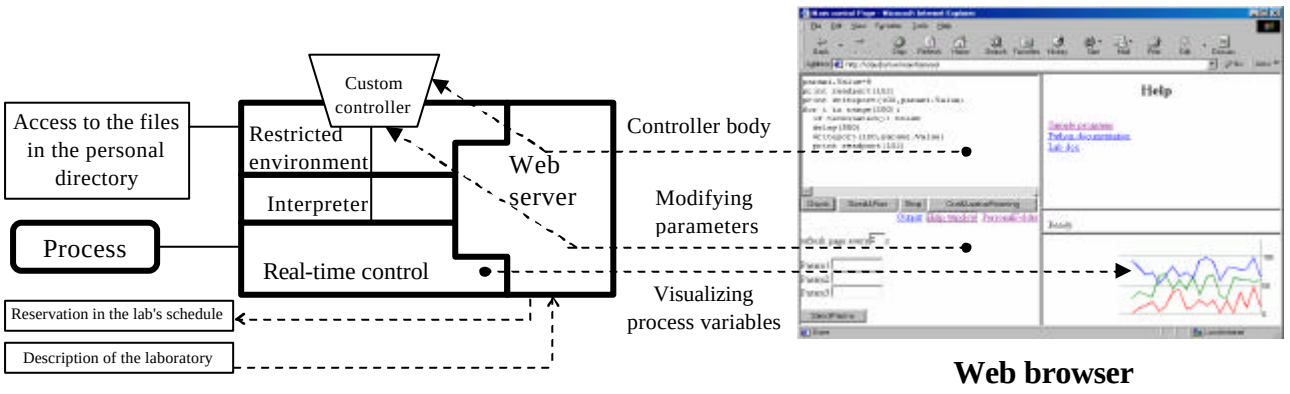

Figure 2: Structure of the server software, including the communication with the client