Network Virtual Laboratory for External Devices Programming

Ryszard Klempous[1], Jan Nikodem[1], Tomasz Walkowiak[1], Jerzy Rozenblit[2]
[1]*Institute of Engineering Cybernetics, Wroclaw University of Technology*
ul. Janiszewskiego 11/17, 50-372 Wroclaw, POLAND
ryszard.klempous@pwr.wroc.pl, jnikodem@ict.pwr.wroc.pl, twalkow@ict.pwr.wroc.pl
[2]*Dept. of Electrical and Computer Engineering*
The University of Arizona, Tucson, AZ 85721, USA
jr@ece.arizona.edu

Abstract

The paper describes an approach to replace traditional computer laboratories by Internet virtual ones. The purpose of this laboratory is to make available advanced learning contents in a network environment. Devices available for students are fully simulated by computers. Furthermore, remote access to programming environment is implemented, which gives the user an ability to write programs controlling a virtual device. The paper describes all elements of this system.

1. Introduction

One of solutions for the global education is the Internet-based distance learning with a usage of modern multimedia technologies. Distance learning is being more willingly and widely employed. Therefore, in an effort to diversify this form of sharing knowledge, it is common to use various available tools given by the available technology.

The idea of distance learning was used for the first time about three hundred years ago with the first correspondence courses. In the age of computerization, it becomes an even more attractive form of learning. The need of gaining more professional knowledge, shortening the time available for stationary methods of learning (esp. when somebody works) resulted in the development of distance learning methods. This form is also an excellent proposition for handicapped people, who for health reasons cannot attend regular classes in a remote university. It is also a very attractive proposition in education of unemployed people.

The Internet is becoming the most universal tool in human communication. Its widespread accessibility gives us a great number of potential recipients of information. We can safely say that Internet is at present the most interesting and dynamically expanding medium for providing educational content in a truly distributed fashion.

Generally, there are two categories of distance learning: synchronous and asynchronous [1]. Synchronous learning requires simultaneous participation of students and teachers. The main advantage of this method is that contact with other participants is possible in real-time. Example forms of synchronous learning are interactive TV, teleconference, Internet chat, etc. Asynchronous learning does not need simultaneous participation of students and teachers. There is no need to meet together at a certain hour. Students can choose a learning place, time period and the amount of material desired to be learned. Asynchronous learning is more "elastic" than its synchronous equivalent. Various forms of asynchronous learning are e-mail correspondence, video, audio contents or WWW page browsing. The main negative aspect of this learning method is that a student can accumulate too many undone tasks.

In Poland, the use of Internet for distance learning is still at an early stage of development. Internet is mainly used to asynchronously access educational materials. It is often considered as a substitute for a book.

Distance learning is still a domain of academic research. However, development of new techniques of transmitting and processing information opens a way to new and more efficient forms of learning. Research work in this area is done all over the world. One of institutions taking part in the research on using multimedia and Internet in learning is The Wroclaw University of Technology, Poland. A research group at the Institute of Engineering Cybernetics has developed an experimental course using the newest Internet technologies in the framework of "MM-EDU" project [2] sponsored by the European Union.

One of the aims was to integrate various techniques of learning [3]. We substituted a traditional lecture with a video recording. A student (user) is able to see and hear recorded fragments of a lecture. Furthermore, he can observe the lecturer's activities on a virtual whiteboard and read additional notes. Traditional textbooks were substituted by dynamic multimedia presentations. The effectiveness of this form of learning was much enhanced by animations, forcing the user to interact with the presentation. Another task was to invent an Internet form of laboratory activity. This resulted in the development of

Proceedings of the 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS'04)
0-7695-2125-8/04 $ 20.00 © 2004 IEEE

a network virtual laboratory. This laboratory does not exist physically. Devices available for students are fully simulated by computers. The purpose of this laboratory is to make available advanced learning contents in a network environment.

Virtual laboratory is an idea developed for many years by large number of researchers [4]. Currently large number of software packages, the most popular are MATLAB and LabVIEW, allows creation of laboratory environment with an access simulations and real equipment via a remote client web interface. Some exemplar application could be found in [5], [6]. The other popular approach is to design simulator of a device, mainly using Java language [7], [8]. The other idea used in virtual laboratories is the distance access to programming environment [9], [10].

2. Network Virtual Laboratory

Network Virtual Laboratory (NetVL) is a computer aided learning system. The aim of the NetVL is to support learning of programming such types of external devices as a plotter, milling table, LCD or an industrial robot. There is also a possibility of programming microprocessor devices such as PCs. The NetVL is based on functional simulation idea [11].

The main purpose of the system is a remote access to those devices through the Internet. The user is expected to have only a Java enabled Internet browser and of course, Internet access. The devices themselves can be real (e.g., a PC) or virtual (e.g., a simulated plotter or thermometer).

In the NetVL environment, peripheral devices are programmed in C/C++. Communication with them is possible through a hypothetical interface, which uses a set of I/O addresses in a virtual PC. Access to those I/O ports is available by the use of C/C++ functions which read/write words into/from desired I/O address.

The user can see the controlled device in a window of the browser. Usually, in the same window there is a module presenting the state of the device's registers that can be accessed from the user's program. There is also a possibility to manipulate individual bits, which can be useful in the beginning of the learning process.

Beside the ability to view the simulated device and its registers, the user can operate in the programming and running environment. He can write programs controlling devices compile and run them without having any compiler installed on his machine. Access to the compiler and the computer, where the user's program actually is being run, is possible through the Internet. Like in the case of inspecting the simulated device, only a Java enabled browser is required. A Java applet, which provides communication with the compiler, runs in the browser's window.

NetVL provides authorized access to the system. The user has to login to the system by entering his name and password. Only after the successful login, can he access all devices and tutorials.

The entire system was written using the Java programming language. This ensures hardware platform portability. We developed a detailed specification [12], which allows the development of a large number of virtual devices that could co-operate.

The NetVL was designed to make available the following functions:

- *Simulation*: simulation of the behavior of the given device based on control data. Possibility of producing the simulator in various ways is assumed, i.e., using off-the-shelf simulators (for processors for example) and custom-built simulators.
- *Visualization*: visualization of the behavior of the given device based on the data from the simulator.
- *Testing*: makes possible to control the device by exposing control ports for direct manipulations. This feature should enable the user to better understand the properties of a particular device by working with the control ports or by running ready-made control programs and observing the results ("Test&Try" method).
- *Programming*: makes possible to create and run on the device simulator various control programs written by the user.

It consists of the following subsystems:

- *Management subsystem* (logging, accounting, etc.)
- *Virtual Devices*: simulator, port, and visualization.
- *Virtual Microprocessors*: writing programs, compiling (on a remote machine), executing (on a remote machine), controlling virtual external devices, terminal like communication with user, distance debugging of programs.

The Management Subsystem (MS) is the main start point for any NetVL subsystem. All applets communicate with the MS and get the information about where the required subsystem is located (under which TCP/IP port). All subsystems can be placed on any computer on the Internet. In case that the required subsystem is placed on a different server than the main NetVL system, all communication goes through the NetVL server to the required computer. It is tunneled, since applets can communicate only with the Web server from which it has started. Such a highly distributed architecture allows spreading the required machine power among all available computers and is invisible to the user.

The MS also keeps track of all users' behavior logging it to a file. It could be used in the future for a user control, security and customization purposes.

Next, the two subsystems: Virtual Devices and Virtual Microprocessors are described.

3. Virtual Devices

The first objective of this module is to teach students how some external devices are built and how they can be controlled by means of digital ports. Each device is visible to the external world like a set of ports, which can be set or read. After learning a module describing a given device, the user will be able to work with the device by clicking bits in control ports and observing results in the virtual device visualization applet.

Internal structure of Virtual Device is presented on Fig. 1. The following are the subcomponents of the Virtual Device (see Figure 1.a):

- *Virtual Ports*. Module that represents (communication) ports in a real control computer (but are accessed through the net). Acts as an interface between the user (VM and Manipulator) and the Simulator. Some of the ports function as input (can be written to) and some as output (can be read). This runs on the server side.
- *Internal Ports*. The same functionality as the Virtual Ports, but having the additional information of the internal state of the device. It sends (on-line) info to the Visualization module that visualizes the device's functionality. This module runs on the server side.
- *Simulator*. Module that simulates the device's behavior. This module runs on the server side.

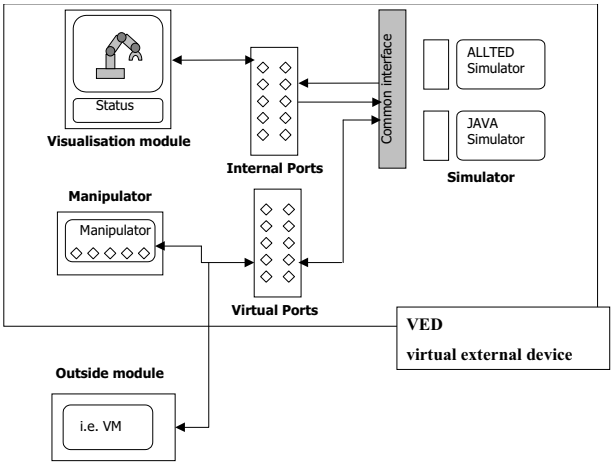

Figure 1. Internal structure of virtual device

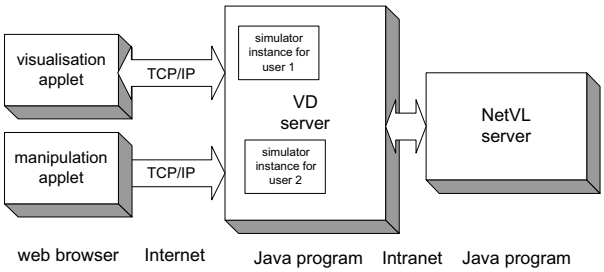

Figure 1.a. Virtual device communication schema.

- *Manipulator*. Gives the ability to control the device (Simulator), by simply playing with the input ports in Virtual Ports (via the Manipulator). This module runs on the user's computer.
- *Visualization*. Module that visualizes the behavior of the device based on the data written to the Internal Ports. This module runs on the user's computer.

The second objective of this module is teaching students how to write programs that control external devices (see next chapter).

The following Virtual Devices have been developed:

- Scrapper (Fig.2);
- Robot,
- Plotter (Fig. 3),
- Heat meter.

4. Virtual Microprocessor

The Virtual Microprocessor (VM) subsystem of the NetVL is focused on teaching how to write programs on a selected microprocessor.

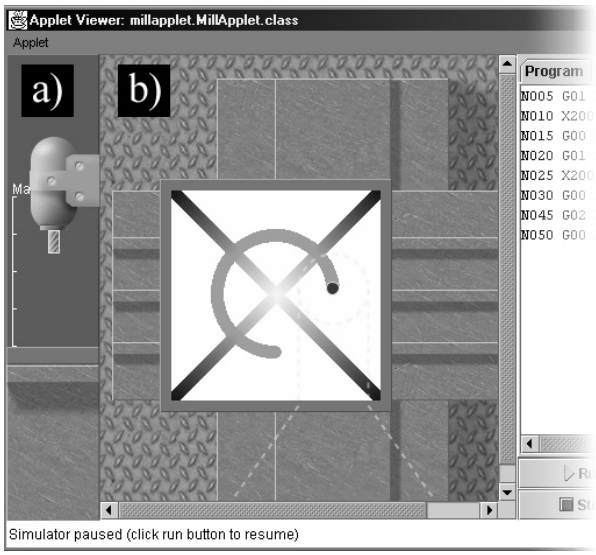

Figure 2. Example of virtual device: Scrapper

Figure 3. Example of virtual device: Plotter

Currently, PC and Transputer computer architectures were implemented. This subsystem familiarizes users with a microprocessor-programming environment. The main tasks of a user are to write programs and observe the results.

The VM is based on the Client-Server idea. A user applet connects to the Management Subsystem (MS) and gets a VM address (TCP/IP port number). All currently built VMs are physically placed on different servers to the MS, so the user applet communicates with the VM through the MS. After making a connection with the VM a user applet authorizes the user (Fig. 4), based on user ID. Next, a user is logged on to the VM subsystem and its applet allows work with files that are stored on the remote server. The user cans (Fig .5):

- create a new file or directory;
- edit and save an existing file;
- change a file or directory name;
- delete a file or directory.

All users have their own disk space that can only be accessed by them (and of course by the administrator). When an authorized user accesses a given VM for the first time, the VM server automatically creates a user disk space.

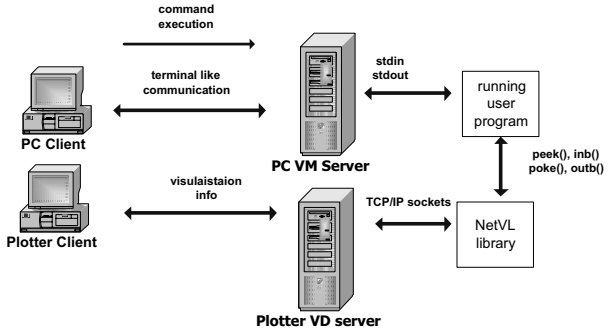

Figure 4. VM authorization procedure

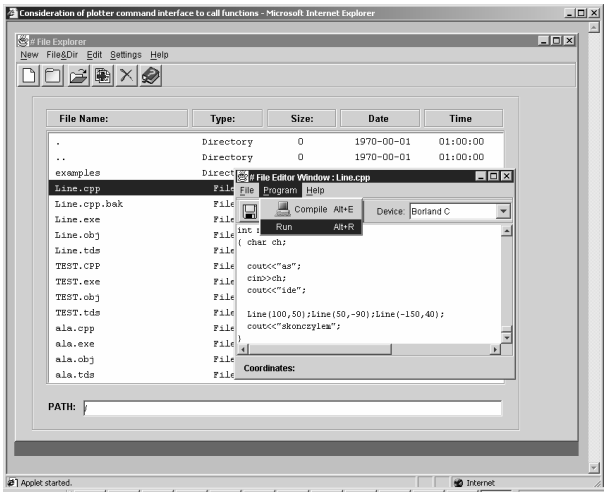

Figure 5. Virtual programming environment

A user can create executable programs. The user applet in the VM subsystem allows the user to run compilers or linkers on remote machines and to see the program outputs on a terminal-like window. Programs can be run on a remote machine and can communicate through a terminal-like interface with the user.

4.1. PC's Virtual Microprocessor

The main objective of this module is to teach students how to write C or C++ programs that control some external devices (such as a plotter, robot, etc.) The PC VM is based on a C++ freeware compiler. It allows the user to write any C or C++ language program (though only terminal user interface is supported) and run it. Using a terminal window, the user can communicate with a program running on a remote computer.

The PC VM also has the ability to write programs that control Virtual Devices (VD). We have developed a special C library giving a user two functions (peek () and poke ()) that allow reading or writing to a VD port. Writing to or reading from a VD port is performed by a TCP/IP connection with the VD server. The visual effects of controlling the VD can be seen in the client applet.

4.2. Transputer's Virtual Microprocessor

The main objective of this module is to teach students how to write Occam programs implementing selected parallel algorithms. It gives an access to real transputer board with four Inmos T805 processors. These VMs allow writing its own program in the Occam language as well as compile, link, and executing it.

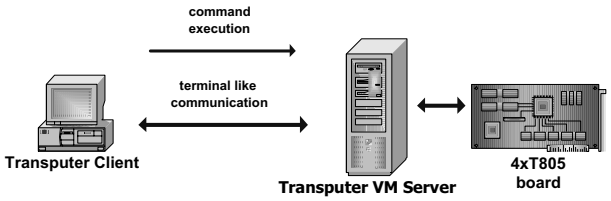

Figure 6. Transputer VM

The main difference to other VMs is that compilation, linking and execution is performed on a real transputer board (not on a PC). The transputer board is fitted in the PC computer with Linux operating system. Thus, this architecture differs a bit from others, i.e., the Transputer server is written in the C language, when others are implemented in Java. However, from the user point of view, this VM is similar to others. The communication with the user applet is also terminal like.

The main advantage of this module is an ability to test real time aspects of parallel algorithms. There is a possibility to run programs on a number of different transputers, e.g., from 1 to 4 to observe the speed-up parameters.

5. Internationalization

A multilingual version of the NetVL packet was realized to enable easy modifications of currently existing modules and quick addition of any language versions. Internationalization encompasses both sides of NetVL package (Client-Applet and servers of virtual devices). All modules have appropriate files with language resources "FileName_xx.properties", where "xx" is a unique country code (specified by ISO Language Code). The internal structure of files with language resources is very simple. Appropriate strings of text in a given language are selected and displayed inside of the files. Identifiers must end with the equation sign after which a string begins in a given user language. All strings with national characters must be written in the 8-bit Unicode standard (UTF-8).

6. Validation tests and conclusions

The system was tested on a group of students. They accessed the laboratory from different sites connected via the Internet and over dial-up lines. When students used the simulators the main problem was to get just right level of interaction, enough that they understand what is happening, that they get feedback about their actions. Too much assistance or feedback was unfriendly but too little was unsatisfactory. It seems necessary to propose a consistent, appropriate image of virtual device that allows the user to construct a mental model of what is going on. This is a really big challenge because each of the students represents different levels of knowledge and skill to realize the exercise, so they required different kinds of assistance and feedback from our network virtual laboratory.

In the actual stage the proposed system is not as responsive as a conversational human being partner, so we must still work on. Because of that, after we have obtained virtual laboratory, now we should learn the moderation in its usage. Creative learning process needs a gentle balance between a network virtual laboratory and a student operating it, as well as direct contact with human being teacher.

The developed distance learning laboratory system is very useful for teaching students how to control external devices. It is available for students equipped with a computer and Internet access and Java plugin. It works with virtual devices (like plotter or robot) as well as with simulated ones (transputers). Tests proved that ordinary phone-line modem access is sufficient for effective usage. All the software has been written efficiently and computing resources are not heavily challenged. The whole system or a part of it could be included in the Learning Management Systems (the test were done for WebCT [1][2]).

Further work will focus on developing new virtual devices, like Heating System Control. We also want to upgrade the robot VD by a usage of 3D graphics with ray-tracing like visualization.

7. Acknowledgement

The work reported here was partly sponsored by the European Union INCO-COPERNICUS project PL96-1046 "Multimedia Education: An Experiment in Delivering CBL Material".

References

[1] "Distance Learning Week: What is Distance Learning?", *Public Broadcasting Services*, http://www.pbs.org/adultlearning/als/dlweek/whatis.htm, 1999.

[2] MM-EDU project http://mm-edu.ict.pwr.wroc.pl.

[3] W. Baranski , T. Walkowiak, "Multimedia approach to distance learning", *First International Conference on Soft Computing Applied in Computer and Economic Environments*, Kunovice, Czech Republic, January 30- 31, 2003, pp. 150-158.

[4] R. Guidorzi, "Abstract tools to deal with reality: virtual laboratories", *Prometeus Inter-SIG Workshop "E-learning for Europe"*, Bologna, September 2000.

[5] A. Bagnasco, A. M. Scapolla "A Grid of Remote Laboratory for Teaching Electronics", *2nd International LeGE-WG Workshop on e-Learning and Grid Technologies: A Fundamental Challenge for Europe*, Paris, France., 3-4 March, 2003.

[6] D. D. Udrea, E. L. Hines, "The Warwick Engineering Virtual Laboratory (WEVL) – Environment and Interfaces", *4th Annual Conference for LTSN-ICS*, NUI, Galway, 28-28 August, 2002.

[7] M. Karweit, "A virtual engineering/science laboratory course", Johns Hopkins University, http://www.jhu.edu/~virtlab/virtlab.html.

[8] M. Guggisberg, P. Fornaro, T. Gyalog, H. Burkhart, "An interdisciplinary virtual laboratory on nanoscience", *Future Generation Computer Systems,* vol. 19(1), 2003, pp. 133–141.

[9] A. Chan, J. Cao, Ch.Liu, W.Cao, "Design and Implementation of VPL: A Virtual Programming Laboratory for Online Distance Learning", *Advances in Web-Based Learning - ICWL 2003, Second International Conference*, Melbourne, Australia, August 18-20, 2003, Proceedings. *Lecture Notes in Computer Science*, 2783, Springer, 2003 , pp. 509-519.

[10] R. Lutticke, C. Gnorlich, H. Helbig "VILAB - A Virtual Electronic Laboratory for Applied Computer Science", *Proceedings of the Conference Networked Learning in a Global Environment*, ICSC Academic Press, Canada/The Netherlands, 2002.

[11] Baranski W., Majewski J., Dobrowolski A., "Functional simulation of microprocessor external devices", *Proceedings of 4-th International Conference Computer Aided Engineering Education.* Krakow, Poland, vol II, September 11-13, 1997, pp.280-287.

[12] Baranski M., T. Walkowiak T., "MM-EDU: network virtual laboratory". *Technical Report*, Wroclaw University of Technology, Institute of Engineering Cybernetics, SPR 2/2001, 2001.