IMTC 2003 – Instrumentation and Measurement
Technology Conference
Vail, CO, USA, 20-22 May 2003

The Next Approach to the Design of a Web-Based Virtual Laboratory

Paweł PYSZLAK, Remigiusz J. RAK, Andrzej MAJKOWSKI,
Institute of the Theory of Electrical Engineering and Electrical Measurements,
WARSAW UNIVERSITY OF TECHNOLOGY
00-661 Warsaw, Pl. Politechniki 1, Poland

Abstract – ***Virtual instruments, as well as networked and distributed measurement systems, are the natural tools, which can be used in a modern didactic process for creating virtual laboratories offered by a group of Universities. In the paper a solution of the experimental model of a distributed measurement laboratory is presented.***

I. INTRODUCTION

The paper summarizes authors investigations in the field of the Web-based distributed measurement systems. They finally lead to the development of a Virtual Laboratory, accessible from a Web page. It can be a very useful tool for distance learning. Virtual Laboratories can be offered by a few of universities. A student can access instruments via a computer network and carry out a real examination of a tested object directly by using a standard Internet browser. The laboratory of tomorrow will effectively help to overcome the barriers imposed by the traditional classroom setting by using an innovative combination of a new approach to learning and the development and application of new technologies. This will introduce a science teaching through everyday experience. Recent developments in virtual instrument technologies, remote access to laboratory and distance learning tools [1],[3],[5],[6],[7],[8] greatly changed the traditional approach to teaching.

In this paper there is enclosed a short description of the designed framework of a virtual laboratory realized as a remote access to distributed instruments and objects. The main goal of the project was to carry out the practical tests in order to disclose the possibilities and limitations of the control software prepared under JAVA environment.

II. THE LAYER MODEL OF DISTRIBUTED SYSTEM

The idea of virtual instrument has strongly influenced architecture of a distributed measurement system based on a computer network. The integration of a measurement system with a computer network makes it possible to create advanced multi-layer information system structures. Such systems, thanks to the developed protocols and interface standards, are totally opened and scalable. The most important implementations are based on a Local Area Network (LAN) which can be, in that case, considered as a kind of "measurement bus".

The organization of the developed system is presented in fig.1. The system is not only distributed in the sense of space but also in the area of management and control. It consists of four layers. Sensor layer is the lowest one. Control layer is the next. As a matter of fact they are together qualified as a classical, but modern, measurement system. Each measurement system can be controlled by an autonomic programmable controller and can have its own communication interface such as RS-488, IEC-625.2, Ethernet etc. The third layer, called system layer, is based on a computer network. It includes application servers, data base servers and stands for service and control of the all system components. Management layer is the last.

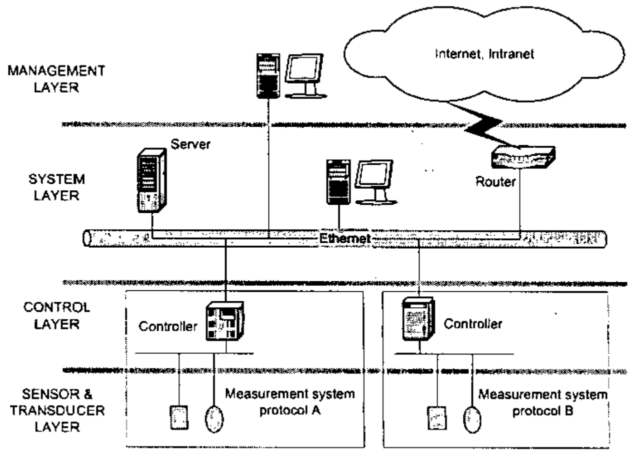

Fig.1 The idea of distributed measurement system

The controllers of the second layer are attached to a computer network, which integrates parts of the system directly, or by the use of special devices, called "gateways". There is growing standardization in the area of complex communication protocols (DataSockets, Industrial Ethernet, Industrial Networking, Java – Jini).

Nowadays the most popular LAN is still Ethernet (simple structure, high reliability, fast transmission speed - 100Mbps, low price). Unfortunately Ethernet was not developed to service the real time systems. It is the most meaningful drawback, from the viewpoint of a measurement system. However the recent update of its specification (IEEE-802), guarantees the maximum delay in information delivering on the level of 4ms [2]. It means that Ethernet can be successfully used in the system layer and in some cases in the control layer.

III. THE IDEA OF VIRTUAL LABORATORY

The idea of virtual laboratory is presented in fig.2. Let us imagine that some research center, say the University A, offers through a LAN some selected resources. All the researchers, scholars and students have limited access to this environment. After connecting the A center to the Internet the offered instruments can be also used by a B center students. The resources can certainly be shared on much wider scale, from both sides. It is very important if we consider a very specialized, expensive equipment which otherwise would not be available to each research center.

Certainly, the most important usage of a virtual laboratory is focused on distance learning. On-line experiments give the possibility to have an impact on a real process or object. The advantages of virtual laboratory are exposed in quite a number of papers [1],[3],[4].

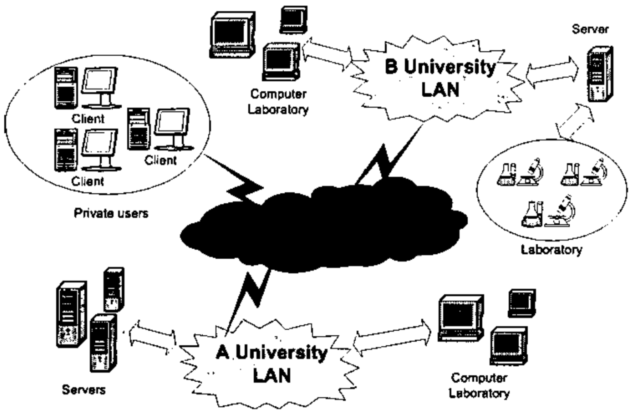

Fig.2 The idea of Virtual Measurement Laboratory

IV. GENERAL ASSUMPTIONS OF THE PROJECT

The software destined for system supervision should implement the following tasks:

- communication process (between users and the laboratory),
- access to laboratory resources (measuring systems and instruments),
- management over the laboratory resources (single instruments or groups of them),
- organization of the users (groups, rights to resources, changes of rights, priorities),
- control over the single users and groups (authentication, authorization),

The measurements should be realized in two modes: "on-demand" and "on-line". The first mode includes two separate cycles: "*query cycle*" and "*answer cycle*". At first a user describes the required parameters of a measurement, next sends a request for making the measurement and sending back the results. In "on-line" mode the user has constant access to the instrument (on-line selection of functions and parameters, watching results).

Software, the most important part of the project, consists of two main parts: server application and client application. Each client includes control panels of a few virtual instruments. A client can be attached to the server - a gateway to real instruments. After login, a session is opened for programming instruments and receiving measuring data. Additionally the server controls rights, enables concurrency of processes, makes multi-access and much more.

V. SYSTEM ARCHITECTURE

The system offering remote access to laboratory has been designed on the basis of a single central server. The architecture of the system is presented in fig.3

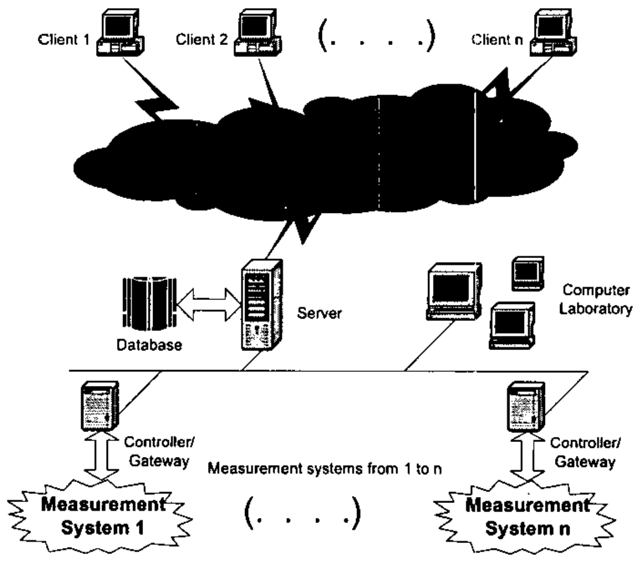

Fig.3 Architecture of the proposed system

There are a few measurement systems which are controlled by personal computers. Each of these computers plays also a role of a "gateway", which enables distribution of measuring data to the single central server. Next, the central server distributes data, using Internet, to every client. Each client can send control data (commands) directed to specified measuring instruments of a selected measurement system.

The architecture, based on a single server, is very convenient. It releases clients from all the controlling processes (access control, multi-access and so on). The client only have to acquire and send data to a server. Such a process does not need a productive computer with complex software. Any excess of the efficiency here can be used for data compression or coding. On the other hand, a really high efficiency is demanded for the central server, especially when a large number of clients is connected to it. In that case a meaningful role plays also the configuration of the local network, servicing the attached measurement system.

Like in a real life, both the users and instruments are divided into groups. All the necessary information about these groups is stored in a specially designed database. The groups of users can have different access rights to control instruments and to receive measuring data. However information about available instruments is stored in the inner structures of the software, but not in the database. The server is able to manage quite a number of measurement systems and quite a number of clients at the same time.

VI. COMMUNICATION FRAMEWORK FOR THE PROPOSED SYSTEM

An efficient communication framework is very important for the overall performance of the presented system. It should be able to service several measurement systems and simultaneously many clients connected to them. Communication procedures should enable sending measuring data and messages among different parts of the system. In this project UDP (User Datagram Protocol) is used for sending measuring data in real time and TCP (Transmission Control Protocol) for transmission control (exchanging messages and commands), and for sending measuring data in the "on-demand" mode. An access to the transport layer is organized through the sockets. The mechanism of properly coordinated threads enables simultaneous work. Figure 4 depicts such a communication framework.

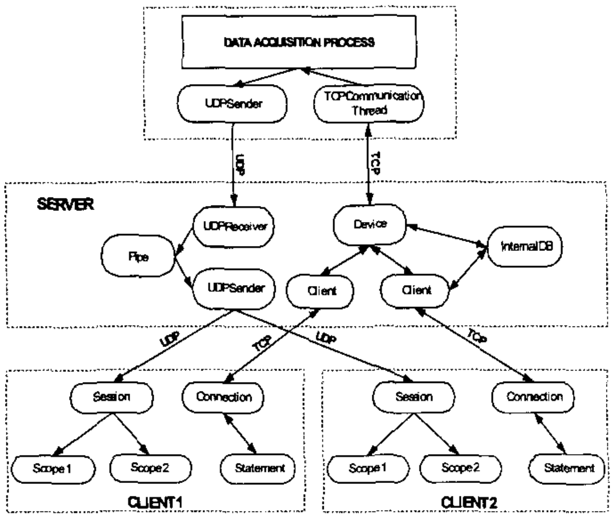

Fig.4 Communication framework for the proposed system

In the figure there are presented the objects (Java classes) that play a key role in data transmission. Also the all possible data transmission directions and the protocols used for transmitting data are shown.

When the server starts an object `Services` is created. It does not take part in a data transmission, but consists of two threads waiting for request of connection from clients or measuring devices. A data transmission is realized just after accepting by `Services` requests from a user or a device and creating associated with them objects. Each connection between a measurement system and a client is realized through two synchronized channels. The channel which transmits measuring data packets is unidirectional. A transmission through it is started by a measuring device and goes through the server to a particular user. The software that controls the measuring device creates a thread which aim is to send packets containing measuring data to the server. This thread sends datagrams to server object `UDPReceiver`, which transmits them to the object `UDPSender`. The datagrams are exchanged between `UDPReceiver` and `UDPSender` by pipes. In this project pipes are used for sending objects (references) rather then bytes what is much more efficient. `UDPSender` is responsible for data packets distribution to clients. From the clients part, the datagrams are received by objects `Session`, which next distribute them among consoles of virtual instruments that are parts of the client software.

Channels using TCP are bidirectional. The communication through them is realized in the manner "question - answer". Messages and commands from clients (for example a set of parameters for a measuring device) are transmitted through the object `Statement` to the object `Connection`. The object `Statement` is also responsible for directing the answer for a particular request to exactly that part of software which sends the request. Next the object `Connection` transmits data to an associated with the client server object `Client`. `Client` makes the decision if the request is directed to the server or a device. If the request is directed to the server, the object makes the demanded operations and sends back the results. If the request is directed to a device, it is transmitted to the proper object `Device`. Because many clients can simultaneously send requests to the same devices, all messages between objects `Client` and `Device` are stored in a FIFO queue.

For the needs of the presented system two communication protocols were developed. The first JMSDP (Java Measurement System Datagram Protocol) uses UDP as a transportation layer and is designed for sending measuring data. The second is a message protocol called JMSP (Java Measurement System Protocol).

JMSDP is a rather simple protocol which enables unidirectional communication. It defines a frame for sending measuring data. The data in JMSDP frame are quantized samples of a signal. Each sample value is a 16-bit number. The length of the packet is 1036 bytes, that stands for 512 samples. Besides the data the JMSDP frame has a header which contain packet numbers, channel number and data acquisition parameters necessary for signal reconstruction by the client (fig.5). Packets are numbered using 16-bit counters. The first counter counts the overall number of sent packets the second one the number of packets associated with a particular channel. Using information provided by the counters the server and client software are able to arrange packets in order and determine how many of them are lost.

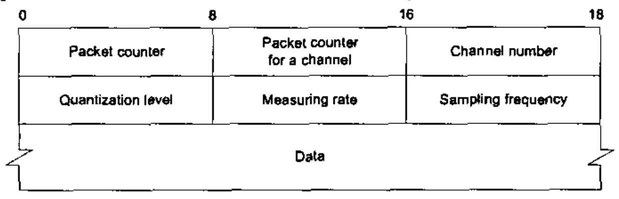

Fig.5 JMSDP frame

JMSP enables exchanging messages that control the whole system. The protocol is based on HTTP/1.1 (Hypertext Transfer Protocol) and RTSP (Real-Time Streaming Protocol). JMSP is a kind of character/stream protocol which organize data in lines of characters. This protocol is designed in such a way that it can also be used to carry binary data sent as so called attachments. The basic unit of JMSP is a message. Messages are divided into requests and responses. Each request demands a response. JMSP is a symmetric protocol what means that both server and client can send requests. It is worth mentioning that the protocol itself does not have to implement commands for controlling measuring devices. On the other hand the devices have to implement the protocol only to the point that enables them connection to the server. Furthermore a protocol that controls measuring devices can be incomprehensible by server software, because JMSP enables a technique called tunneling.

VII. THE STRUCTURE OF THE CLIENT AND SERVER SOFTWARE

The server consists of four software packets named: `core`, `db`, `managers` and `ui`. The `core` packet is responsible for described earlier communications among different parts of the whole system. The `db` packet controls connections to databases describing users and devices. The `ui` packet defines basic user interface. The last packet `managers` enables full management of users, devices, groups and access rights.

The basic aim of the client application is to enable a friendly interface, which makes the resources of the Virtual Laboratory available to a user. The client software was designed in such a way that it resembles a real measuring position, to what several different signals can be delivered. Each signal can be examined using several measuring devices simultaneously. In the system a virtual equivalent of measured signals are JMSDP sessions. The number of possible sessions is not currently restricted, and to each session one can connect any number of measuring devices available from console. The main software packets that builds the client are `ui` (user interface), `scope` (an example of a measuring device) and core. The essential part of the client software `core`, is responsible for communication with server through JMSP and JMSMP protocols.

VIII. CONCLUSIONS

Practical tests and experiments of the project were carried out on different hardware and software platforms with the usage of Java Virtual Machine ver.1.3, both in local and global network. Windows 98/NT/2000 and Linux RedHat 7.2 were used as operating systems.

Implemented computers had a constant access to the Internet throughout the wide-band links. In the majority of tests the developed software worked properly with the sufficient productivity. All the problems have been caused by the environment configuration. The most important one appeared in a case when a client was attached to the Internet throughout a "firewall". In that case the UDP will need special widening or even should be replaced by the other type protocol (RTP).

REFERENCES

[1] L. Hesselink, D. Rizal, E. Bjornson: *„CyberLab: Remote Access to Laboratories Through the World-Wide-Web"* International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet, July 31-August 6, 2000, L'Aquila, Italy.
[2] K, B. Lee, R. D. Schneeman: *„Distributed Measurement and Control Based on the IEEE 1451 Smart Transducer Interface Standards"*, IEEE Transactions on Instrumentation and Measurement Magazine, Vol 49, NO. 3, czerwiec 2000.
[3] A. Ferrero, V. Piuri: *„A Simulation Tool for Virtual Laboratory Experiments in a WWW Environment"*, IEEE Transactions on Instrumentation and Measurement Magazine, Vol 48, Nr. 3, czerwiec 1999.
[4] L. Benetazzo, M. Bertocco, F. Ferraris, A. Ferrero, C. Offelli, M. Parvis, V. Piuri: *„A Web-Based Distributed Virtual Education Laboratory"*, IEEE Transactions on Instrumentation and Measurement Magazine, Vol 49, Nr. 2, kwiecień 1999.
[5] R.J. Rak, *"Virtual Instrument – the main part of Internet based distributed system"*, International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet, July 31-August 6, 2000, L'Aquila, Italy
[6] R. J. Rak: *„Modern Tools for Development and Design of Virtual Instruments"* Third International Conference on Enterprise Information Systems Proceedings vol.2, Setubal, Portugal, July 7-10, 2001 pp.1166-1169.
[7] B. Galwas, R.J. Rak, *"Virtual Laboratory - A Future Part Of The New Web-Based Model of Undergraduate Engineering Studies Developed By Warsaw University Of Technology"*, Joint IMEKO TC-1& XXXIV MKM Conference 2002, 8 - 12 September 2002, Wrocław
[8] E. Michta: *„Communication Models of Networked Measurement and Control Systems"*, Wydawnictwo Politechniki Zielonogórskiej, Zielona Góra, 2000.
[9] G. Coulouris, J. Dollimore, T. Kindberg: *„Distributed Systems Basics and Designing"*, WNT, Warszawa 1998.
[10] W. Richard Stevens: *„UNIX – Programming of Network Services"*, WNT, Warszawa 2000.
[11] D. E. Comer: *„Computer Networks and Internet"*, WNT, Warszawa 2001.