An educational laboratory for remote instrumentation programming based on LabWindovs/CVI™ Environment

Baccigalupi A., Cennamo F., Masi A.

Department of Computer Science - Via Claudio 21, 80125 Naples
University of Naples Federico II - Italy
Ph: +39-081-7683166, Fax: +39-081-7683816
e-mail: alemasi@unina.it

Abstract- The paper describes the architecture of a remote laboratory aimed at training the students of electronic instrumentation graduated courses. The remote laboratory, including a number of measurement stations, each tailored for solving specific tasks, is used as a training environment to develop and test a set of measurement experiments via WLAN and TCP/IP protocol. Once access to a specific measurement station has been granted, it is possible to send commands and receive data from the instruments in order to execute the measurement procedures as if they were performed on a local computer. The communication between the controlled remote laboratory and the local computer has been designed to be totally transparent. Therefore, the measurement devices are controlled in the same way as if they were local instruments. The proposed architecture training target requires particular attention to the access management in order to guarantee waiting times as short as possible.

I. Introduction

The use of laboratory resources is of primary importance for the training programme of electronic measurements courses in engineering faculties. However, the increase in the number of students and the restricted space available for laboratory infrastructures represents a serious problem for this to be guaranteed. Furthermore, the budget available for acquiring the proper number of measurement devices is very often not adequate [1]. The proposed solution is based on the realization of a remote laboratory, including a number of measurement stations, tailored to solve specific tasks, to be used by students as a training environment to develop and test a set of measurement experiments via internet (e.g. WLAN and TCP/IP protocol) [2]. In this way, two main problems are solved (i) every measurement station is time-shared among several students so that fewer stations are required and (ii) laboratories are not overcrowded anymore.
Based on these assumptions, a virtual laboratory oriented to the programming of electronic measurement instrumentation was realized.

II. The proposed architecture

The architecture of the remote laboratory is sketched in figure 1. The instrumentation servers ***"S"*** run on computers that physically control the measurement stations via the GPIB (IEEE-488) bus. The clients ***"C"*** represent generic users (e.g. students) requesting access to the remote laboratory, i.e. accessing a specific measurement station on which they intend to perform a training activity. The heart of the system is the monitor ***"M"*** that manages and coordinates the communication between all the units. It is updated with the status of all the active servers and handles the connection requests coming from all the clients. Furthermore, the monitor allocates the resources to the clients, by properly administrating the access queue and by using the watchdog timer for handling time-out conditions of the connected clients. In particular, the use of the measurement station is limited to a time slot depending on the number of requests waiting for the same resource. In addition, every resource is immediately revoked if so requested by the client administrator. Therefore, every client is equipped with an inactivity timer that starts after every command is sent, and counts its inactivity time. If the latter one exceeds a fixed threshold, the resource is released. The set of these rules was thought in

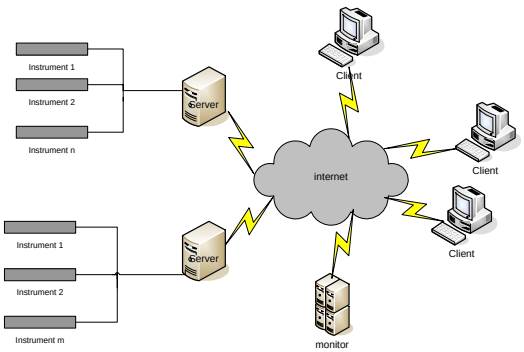

Figure 1: Remote laboratory logic architecture

order to achieve the optimal use of the available resources; in particular it incentives the careful use of resources ensuring, as a consequence, acceptable waiting times even when the number of students is much bigger than the number of instrumentation servers.
When the monitor assigns the resources, a one-to-one client-to-server connection is established. Thus, within the obtained time slot, the client has the complete control of the measurement station so that commands and the consequent responses can be exchanged. In order to reach this goal, a proper library of functions to send commands to the remote instrumentation as well as to read the answers was implemented. The syntax used in the library functions is exactly the same as the one used by the functions of the standard IEEE-488.2 library (e.g. ibwrt, ibrd, ibrdf, ibfind, ibclr, ibloc, and so on). In this way, the application program has the same structure of a program working on a computer directly connected via the IEEE-488.2 instrumentation bus. The remote connection is, therefore, totally transparent to the user. As in the case of local applications, every device is identified by a "device handle" obtained in response to the ibfind command (specifying the device physical address). When user accesses the virtual laboratory, he/she receives the composition of all the types of measurement stations available, so that before making a connection request the physical addresses of the wanted instruments are known.
The remote laboratory has a number of measurement stations of the same type, but this information is totally transparent to the remote clients. The latter one, however, has to be properly handled by the monitor that has to dispatch the requests of same measurement stations on different instrumentation servers, dynamically, taking care of the state of use. The working units of the above sketched architecture were developed with LabWindows CVI™ by using National Instruments Datasocket™ technology. The data exchange channels between the working units of the proposed architecture are shown in Figure 2.
The spy unit is an application example of the proposed architecture. It allows the observation of the selected server working activity. This includes the commands sent by the client to the server and the corresponding responses sent back to the client. This feature is very useful for the teaching goal in that the teacher, at any time, can check the training session of a particular student. The data channels, connecting each server to the monitor, allow real time updating of server status (i.e. server on-line, eventual errors, sending of the instrument handle). In the same way, by means of two additional channels, the monitor sends information to the clients and to the spy connected to the laboratory (for the former ones the measurement station types available, while for the latter ones the IP of the servers that can be monitored). Finally, the last datasocket channel from the monitor to every server is used to manage the attribution of the time slot.

a. The Monitor "M"

The Monitor represents the heart of the architecture proposed. In fig.3 the working state chart of the software module is shown.
M, after launching and connecting to the servers, is usually in a idle state, waiting for the requests from the clients. The generic **C**, by means of writing on the specific Datasocket channel, sends a demand for a particular measurement station jointly with the user name and password of the user. Before assigning

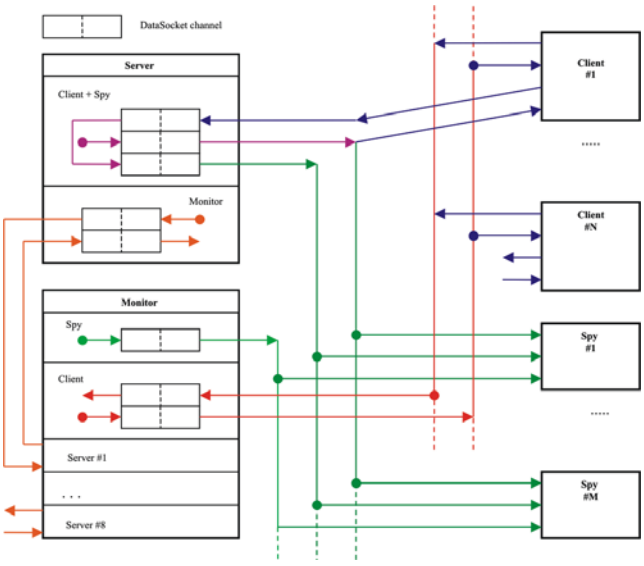

Figure 2: Remote laboratory physical architecture

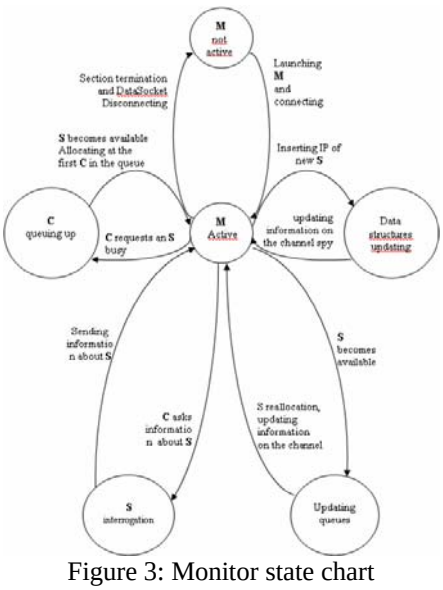

Figure 3: Monitor state chart

the requested resources, **M** checks if the user is present in its local database and, obviously, if the resource is available. If the requested resources are currently busy, the user is put in the queue of the servers connected to the measurement station requested. The resource assignment consists of sending the Ip of the server available to the client that made the request, so that the latter can establish a point to point connection. The account *Guest* is also foreseen, but it is characterized by limitations in the queue access right and in the server using time slot. Only a recognized user holds his priority in the queue even after leaving the laboratory, so that at the next connection a higher position in the queue will be granted when asking for the same station. The client, after gaining the connection with the instrumentation server, has a fixed time slot of 10 minutes assigned if there are other users waiting in the queue. Otherwise, an unlimited using time is allowed unless the administrator revokes the resource. The inactivity timer on the client starts after sending every command and is characterized by a threshold of 1 minute.
M is configured with the Ip addresses of all the instrumentation servers of the remote laboratory, classified as measurement station types. Following a station information request by the generic client, the monitor interrogates the least busy instrumentation server of the requested type, and returns to **C** information about the configuration of the remote measurement station, i.e. the physical address of the instruments locally connected and their identification string. The physical address of the instrument will be successively used to drive the remote instrumentations as already said above.
Another important duty of **M** is to update the Spy units with the servers' activity status, when a server previously busy becomes available or a new instrumentation server is connected to **M**.

b. The Client "C": remote IBIC

In the following, the term "Client" will be referred to any LabWindows CVI application using the resources of the remote laboratory with the aim of programming remote electronic instruments and using the library of the properly designed IEEE 488.2 remote functions. The main characteristic of the developed remote functions is the complete compatibility with the IEEE 488.2 standard functions, so that the user can train to work on the remote instrumentation in the same way as the instruments were locally connected to his computer. An authorized user can convert a local application into a remote one simply by establishing a proper connection with the instrumentation server. In this way, he will receive instrument handles by simply using the remote command "ibfind" along with the physical address obtained by the monitor the monitor **M**. In addition, in the remote communication, the standard variables "status", "error" and "count" are properly handled as well. Specific functions handle the connection and unconnection between the laboratory units: in particular, the function *RequestConnection* requests **M** for the connection to a particular measurement station. When the resource becomes available, **C** receives the server IP and enables a new connection by using the function *ServerConnection*.
To support users in training of electronic instrumentation programming, a particular client was developed: the *Remote Ibic* (fig.4). It can be considered as the equivalent remote release of the standard NI tool that manages interactive control of GPIB instrumentation.
So as to assure a good communication reliability even in the presence of high network delays, the crucial point in the development of this module was the syncronization of the writing and reading operations. The *ibwrite* function is based on sending a command string at variable length plus its cyclic redundancy check (CRC), to the specific server. After receiving the command, the server checks its correctness using the CRC and puts this on a dedicated Datasocket channel. **C** only sends another

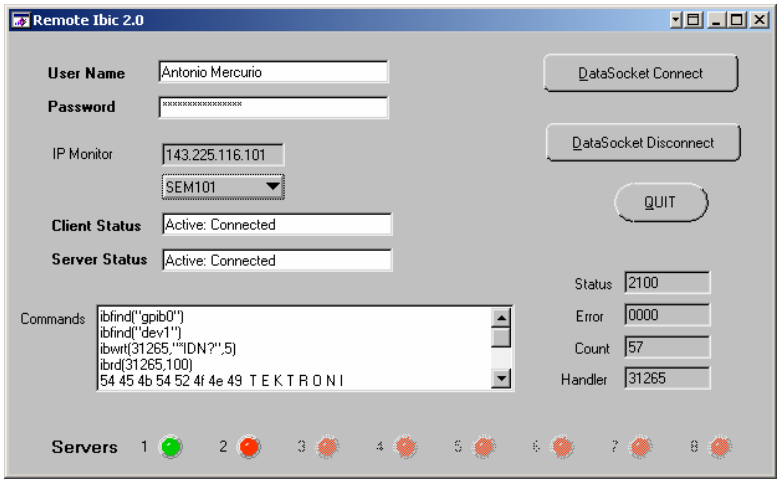

Figure 4: Remote Ibic user interface

command when it receives its previous CRC.

c. The Server S

The module **S** is installed on every computer connected to the proper measurement station. Apart from driving the electronic instruments via the 488 interface, **S** also manages the communication with **M** and **C**. Every instrument on-line is identified by a specific handle obtained *only once* by means of the *ibfind* command executed locally. These instrument handles are stored and sent to every client that gained access to the measurement station. If, for some reason, an instrument goes off-line, **S** updates the monitor with the new status of the measurement station immediately, through the specific DataSocket channel. The same channel is also used to send the configuration information, following a request by **M**. After the connection with a particular **C** is established, all the commands sent with the function remote *ibwrite* are executed, even if they are wrong. The error condition is shown through the variables *ibstatus* and *iberror* received by **C**. Another important feature of **S** is the "forwarding" of the commands from the client **C** to the unit *Spy Ibic*, using a specific DataSocket channel so that the module *spy* can monitor both the commands sent to **S** and its responses.
Presently **M** can manage up to eight different servers. The modularity of the architecture allows other monitor units to be installed whenever a bigger number of measurement stations may be required.

d. The Spy Ibic

The Spy unit represents a valid support for the teacher in the training activity; he can use this module to survey a work session of a particular student connected to a measurement station. In particular the log file, composed of commands sent to the instruments and of responses by the specific **S**, could represent the paper of an examination session.

IV. A laboratory facility demonstrator: ADSIM

In order to evaluate the laboratory perfomance as well as the accessibility of its resources, a client application was developed. It aimed at remotely controlling a measurement bench for the verification of analog-digital converters making use of the client library previously described [3].
The main interface of this client is shown in fig.6. The part devoted to accessing the laboratory is evident. For this application only the measurement stations based on (at least) a function generator (HP 33120 A) and an oscilloscope (TEKTRONIX TDS 1002) are required. In order to choose the test methods for the conversion system, factors like the instruments available, the computation power needed and the instructive value, were taken into account. The dynamic test based on codes histogram evaluation [9], is used to obtain the real transfer function of the oscilloscope converter in addition to integral and differential non linearity error. In oder to evaluate the effective number of bits and the signal to noise ratio the well known method of *sinefit* (with 3 parameters) [6] was implemented. An approach based on the analysis in the frequency domain was used too: the DFT is computed on the pure sinusoidal signal acquired or directly on the residual between the signal acquired and the result of the fit. Particular attention is devoted to the signal windowing before the FFT computation, in order to reduce the spectral leakage effects.
Adsim can be regarded as composed of sections through which the user can control the remote instrumentation (e.g. changing of function generator parameters or oscilloscope setting) and the virtual

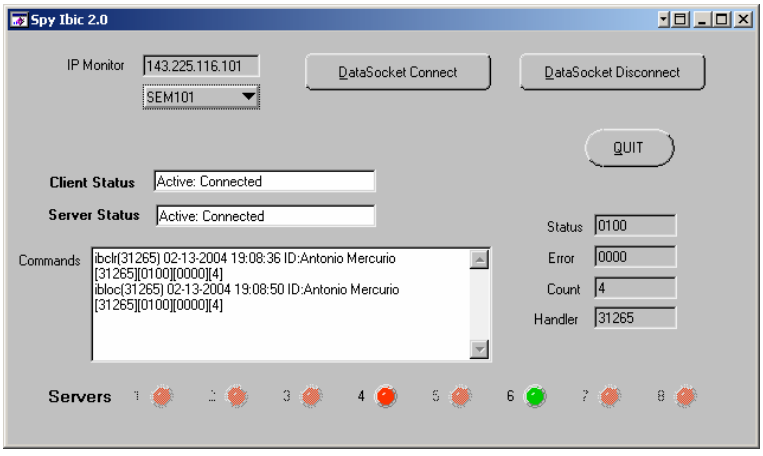

Figure 5: Spy Ibic user interface

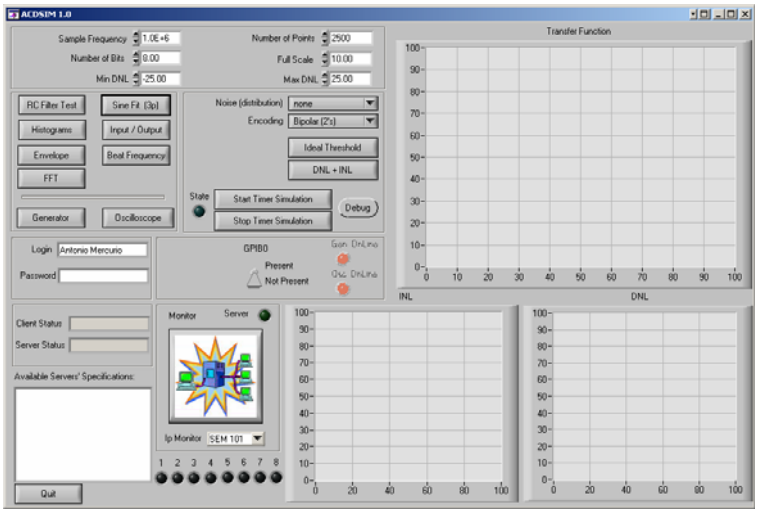

Figure 6: Main panel of the Adsim application

instruments. The latter ones are software tools that process the data acquired on-line by the remote instruments, delivering then the results to the user (all the methods implemented for the converter testing belong to this category). Therefore, keeping in mind the educational issues of this application, a simulation mode is also provided. In this way the student can alternate phases of on-line communication with the instruments and simulated parts so as to better understand the results obtained.

V. Results

Many tests on the proposed educational laboratory were carried out, with the intention to characterize the access time to the resources by the remote clients. These were performed in different network conditions and also with measurement stations set up in different places. A reference configuration composed of two measurement stations (both arranged with a function generator HP 34120 A and an oscilloscope TEKTRONIX 1002) was used. These two latter ones were installed, respectively, at the University "Federico II" laboratory and the Military Areonautic Accademy of Pozzuoli (NA). As **C** the *ADCsim* application was considered, and different client units were installed. The extreme cases of connection to the local network (one through a typical 100 Mbit LAN and the other through a GPRS connection with a typical bandwith of 3-4 kbytes/s), were taken into account. The choice of a resource

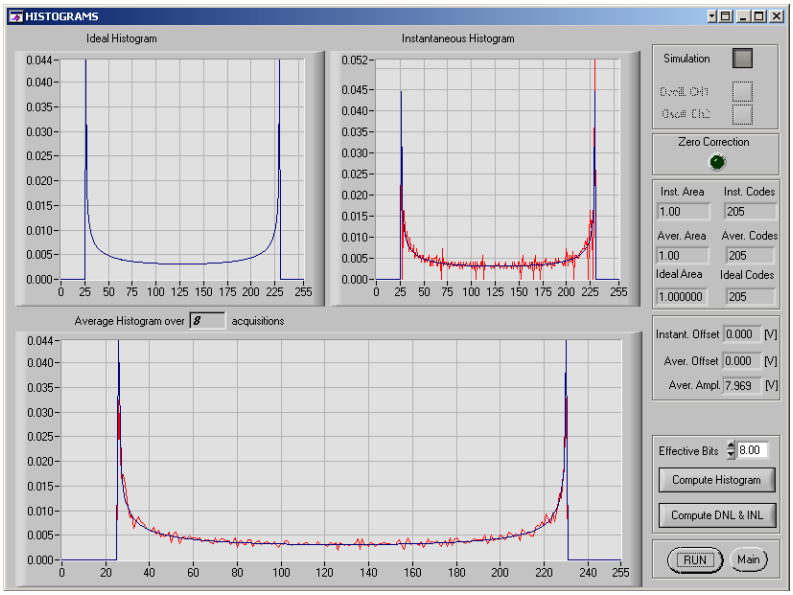

Figure 7: Histogram average and instantaneous relative to a sinusoidal input at 400 Hz frequency and 8 Vpp. To obtain stable results at least 30 asynchronous acquisitions are requested

utilization time slot of 10 minutes long guaranteed an acceptable waiting time for a busy server, even with the number of **C**s connected being much higher than the number of installed measurement stations.
The delay time between the command sent by **C** and the reception of a response (e.g. *ibread* command to acquire a setting of the remote instrument) can become intolerable with a slow connection. Measurements show that it is negligible for a LAN connection, but can be as much as few seconds in the case of a GPRS connection (with a worst case of 25 seconds). The syncronization strategy adopted anyway guarantees that no commands are lost and that a new command is sent only after the previous one was executed. However, in these conditions the working of *ADsim* is critical. After connecting with the remote instrument, a lot of *ibread* commands are sent to obtain the setting of the instruments. This operation requires a few minutes. In addition, all the virtual instruments that need data coming from repetitive acquisitions by the oscilloscope (like the histogram methods (fig.7)) will be characterized by very long computational times. The same delay time can be considered to be acceptable for a home connection using a 56 kbaud modem.

VI. Conclusion and outlook

The architecture of the educational laboratory proposed has shown robusteness and reliability, as well as short times for client access and working, using a university LAN connection. Both the application *Remote Ibic* and *Adsim* represent a valid training support. Now the teacher can use this facility to illustrate "live" the concepts of electronic instrumentation programming and the proper functioning of the instruments. Finally, using *Adsim*, the results in real time and on a real conversion system of the A/D test methods can also be illustrated [4]. The crowding problem of the instrumentation laboratories during the students training sessions is solved. The student will access a remote client terminal, use the remote IEEE 488.2 library and develop his software within LabWindows CVI enviroment, debugging it as if working locally on the instruments. Of course, the client terminal can also be installed on the same site as the measurement instruments to give the student the opportunity of accessing the measurement instrumentation physically.
To date, the only problem observed is the long reacting time of the remote instruments using a slow network connection (e.g. GPRS). This is the consequence of the approach followed in the client applications development which has to be the same as the approach used for a local instrument program, for compatibility reasons. In fact, to obtain the remote instrument status, a command has to be sent every time, whereas a block mode transfer would assure lower dead times.
Following this idea, we are developing other clients modules that drive every single remote instrument and are characterized by a data transfer optimization in order to increase the data exchange speed, even with a slow network connection. These modules can be seen like drivers to access in efficient way the remote laboratory resources and show a user interface very close to the phisical instrument one. Students will use these modules when they want to utilize remote laboratory instruments, changing setting and acquiring data on which data treatment will be performed [5].

Acknowledgements

The authors wish to thank Ing. Antonio Mercurio for his useful contribute in the software development.

References

[1] P. Arpaia, A. Baccigalupi, F. Cennamo, P. Daponte, "A Measurement Laboratory on Geographic Network for Remote Test Experiments", *IEEE Trans. on Instr. and Meas.*, vol.49, n.5, pp. 992-997, 2001
[2] P. Arpaia, A. Baccigalupi, F. Cennamo, P. Daponte, "A Remote Measurement Laboratory for Educational Experiments", *Measurement*, vol.21, n.4, 1997.
[3] P. Arpaia, A. Baccigalupi, F. Cennamo, M. D'Apuzzo, "Remote Characterization of Scan Converter based Transient Digitizers", IMTC. IEEE Instrumentation and Measurement Technology Conference on Sensing, Processing,, 1997.
[4] P. Arpaia, A. Baccigalupi, F. Cennamo, P. Daponte, "Ethernet Application Perspectives for Distributed Measurement Systems", IMEKO. XIV World Congress, 1997
[5] M. Cobby, D. Nicol, T. S. Durrani and W. A. Sandham, "Teaching electronic engineering via World Wide Web, Proc. Of IEE Colloquium "Computer Based Learning in Electronic Education" pp.7/1-11, 10 May 1995,London UK,
[6] IEEE Standard for Digitizing Waveform Recorders (IEEE Std 1057-1994)