Remote Data acquisition and Instrument Control Using Labview and Appletview

O.F. Toader*

* Name Nuclear Engineering and Radiological Sciences, University of Michigan, Ann Arbor, MI USA

***Abstract*—Having the capability to control and observe experiments from a remote location has several benefits, including the ability to track and to assist in solving a problem that might arise. The best way to remotely control and monitor activity during an irradiation experiment is via the World Wide Web, using a method that is browser and computer independent and with software easily available on the market. The Michigan Ion Beam Laboratory (MIBL) at the University of Michigan in Ann Arbor has developed the remote monitor and control capability by using a combination of commercial software packages (Labview and Appletview). During an experiment, all the parameters could be accessed and monitored remotely by simply reaching a web site. A specific description will be given of the way in which this procedure was implemented at MIBL.**

***Index Terms*— Remote engineering, Appletview, Labview, MIBL**

I. INTRODUCTION

The Michigan Ion Beam Laboratory (MIBL) is located in Ann Arbor and is a part of the Department of Nuclear Engineering and Radiological Sciences at the University of Michigan. Much of the recent work done in the laboratory is related to the radiation damage experiments in metallic alloys, which can require significant radiation time (from a few hours to two weeks). Having the capability to control and observe from a remote location has several benefits, including the ability to track the experiment and to assist in solving a problem that might arise. A significant fraction of the work is done in collaboration with researchers from different organizations or institutions necessitating a means to track the progress of the experiments remotely. The best way to remotely monitor an experiment is via the web with software that is platform independent

II. LABORATORY DESCRIPTION

The laboratory is equipped with a 1.7 MV Tandetron accelerator, a 200 KV ion implanter and an ion beam assisted deposition system (IBAD). Remote monitoring could be implemented only on the Tandetron and the IBAD system as the implanter does not have advanced computer interface. The accelerator is a solid-state gas insulated, high frequency device, capable of operation between 0.2 and 1.7 MV with a Torvis-type source. The Tandetron has two beamlines, a 15^0 beamline for ion beam modification (implantation, mixing) and radiation damage, and a 30^0 beamline for ion beam analysis, each terminated with a target chamber. Both beamlines contain a quadrupole magnet for focusing, an analyzing magnet, a raster-scanner and a steerer. The components that are interfaced and controlled/monitored by computer are the Torvis source and the end-station data-collection system. The control/monitor software programs are written in Labview (National Instruments) and were developed at MIBL The IBAD system consists of two 6 kV electron beam guns and a low energy Kaufman ion gun in a large cryopumped vacuum chamber with a base pressure in the order of 10^{-9} Torr range. The system is using for controlling purposes GPIB and serial lines interfaced with a computer. The software running on the computer to monitor and control the deposition experiments is also written in Labview and the code developed at MIBL.

III. COMPUTER HARDWARE AND SOFTWARE

A. The Hardware

Most of the computers in MIBL are 2.5 GHz Dell computers. The computer controlling the Torvis source is interfaced to the ion source through a serial link and a set of analog and digital modules placed on a network of motherboards of type Ioplexer, manufactured by DuTech Inc. The modules allow for voltage and current input and output, as well as digital levels of input and output. The commands are text based from the computer via serial ports. The computer monitoring the stage parameters has a NI PCI temperature board, a NI PCI analog/digital board and a NI scope card. The communication with the end sensors (temperature, voltage, chamber pressure) is provided by a Labview program. The computer controlling the IBAD instruments has an analog/digital board (NI PCI 6024) and a GPIB card. The communication with the instruments and the end sensors is made through a Labview program via the analog/digital channels, GPIB and serial protocols.

B. The Software

In the quest to develop the remote capability several options were considered:

- Data Sockets

The data socket (DS) is a unified end-user application programming interface (API). It consists of a data socket

server (DSS) and an API. The advantages of this route are the reliability of the program, the nice Web interface (all the Labview graphical user interfaces (GUI) configuration tools available) and the speed of the data update. The main disadvantage is that in order to be able to view the experiment the web user has to download and install the Run-Time Engine version of Labview (from NI) plus a small Labview executable program written at MIBL, or download the entire application as a self-installing program from the MIBL site. The size of the download of the whole application is rather large (> 10 MB), making the process a little difficult for slow computer. In addition, the program needs to be recompiled on a Macintosh machine to have it available for Macintosh users.

- Labview built-in server

Web monitoring with the Labview Web Server was implemented after Labview 6.0 was released. Now version 7.1 is out and it has even better remote access features. Starting with the 6.0 version, any application could be published as a web file rather easy using a built-in Web Server. The Labview Web Server approach has very nice security and user monitoring features, but also some disadvantages. Like the previous method, it would require the user to download and install the Labview Run-Time Engine before the first run. The Labview Web Server comes with a built-in one-client capability, limiting the number of users. Additional licenses can be purchased, but cost could be an issue with this strategy. But more important, this alternative is not available for Macintosh computer users. However, the simplicity of the approach and the requirement of very little extra programming besides the main application make this option really appealing.

- Labview/Appletview**.**

With Labview one can connect to other applications and share data through ActiveX, the Web, DLLs, shared libraries, TCP/IP etc. In addition to Labview, the other software package required was Appletview (Java based), available from Nacimiento Inc. Appletview assists in publishing the Labview instrumentation on the web and makes the web browsers capable of connecting to the applications running on a local computer. The built-in server mediates the communication between the local and the remote computer, as the instruments' outputs are made available to a remote client. This is accomplished by streaming the data from the local instruments to a Java applet running in a Web browser. The software enables the programmer to configure and shape the user interface according to the needs. The most important advantages of the Labview/Appletview combination are: the end user has access to DAQ systems located anywhere, is language independent and operating system (OS) independent. Using this protocol, data can be sent via the Web between different machines running Labview.

IV. LABVIEW AND APPLETVIEW

A. Labview

Labview is a powerful software package that can be used to design user interfaces to interactively control the systems in a graphical environment taking advantage of many programming tools and by creating virtual instruments that mirror real ones allowing remote users to collaborate in real time [1]. This software package is produced by National Instruments and has world wide acceptance and presence at this time. It can be found in most of the research labs and in very many research and development facilities of private businesses. It provides a quick and easy access to instrumentation control and a very large database of drivers for DAQ cards, various computer interfaces (GPIB, serial etc) and instrument drivers[1]. Each Labview program consists of a Front Panel (FP) interface, that contains the controls and data fields, and a Block Diagram (BD) where the real programming flow takes place. Although it is a graphical programming language, its versatility allows the easy incorporation of modules written in other languages (C, Basic etc)

B. Appletview

Appletview is based on three main features [2]:

- It uses messages partitioned in groups called channels as a way to synchronize objects across networks. A channel is a logical grouping of objects. A good analogy is a „chat room" where a client connects to a given channel in order to be able to communicate information.
- In order to establish a connection, a client must receive an authenticating „token" (static or dynamic) that is integrated into the website security and is required to allow access to the data.
- It is object oriented. Objects consist of their membership groups, their types and their properties allowing for a unique graphical look and user permissions for any given client of an Appletview system.

Any message submitted and any message received by a client must be sent/received through a server. All clients are considered peers but with distinct permissions to access and receive some types of information. This message system is above the Application Layer (AL) level, higher that the TCP/IP protocols being capable of running across HTTP, FTP, UDP and other networking protocols. An XML language called VIML is used to express the token information[2]. What is great about this is that the information stored in a VIML file is platform independent and could generate an interface for web browsers, hand-held computers, cell phones and so on. These are the minimum requirements in order to run Appletview on a computer in „Server" mode:

- Labview 6.0 or higher
- TCP/IP protocol installed on the server computer

- About 8 MB of HD space for the Appletview VIs.
- Java JRE version 1.4 or higher

As a „Client“ the minimum requirements are:

- For Windows: 95 and up, Netscape 4.05 or higher, Explorer 4.01 or higher
- For Mac OS: Apple Safari, Netscape 6 or higher, Explorer 4.0 or higher
- For Unix: HotJava 1.1, Netscape 4.05 or higher, Explorer 4.01 or higher, Mozilla.

V. HOW IT WORKS

In order to make the connection to the Web, it is necessary to build an application with Labview that would collect the data points of interest from the main DAQ program. The data objects to be sent out are only allowed to be of a certain type (string, integer32, single and Boolean type). A Labview Server built with Appletview libraries (figure 1) would broker the data transfer between Labview DAQ program, the newly built Labview application and the built-in Appletview server running in the background. The VIML application interfaces are built in Appletview with an application called AppletBuilder (Figure 2). Next, the VIML interfaces built with the AppletBuilder are incorporated into a web page as applets. Although as seen in Figure 2 there is a limited pool of available controls to build this interface, for all the applications required at MIBL, they proved sufficient.

VI. EXAMPLES

Next are presented two of the program interfaces that were created in the process of control and monitor irradiation experiments at MIBL. First one shows the application built to run and control the ion source. The screen that is available to a user on the server computer (in the lab) is shown in figure 3 and the web interface available on a client computer (remote) that allows full monitor / remote capabilities of the source is shown in figure 4. Another server / client comparison is shown in figures 5 and 6. These represent a local (figure 5) and a remote (figure 6) view of the application running on the stage monitoring computer. A part of the back diagram showing the connection with some of the Appletview VIs is presented in figure 7.

The system using Labview/Appletview combination is reliable and it can be viewed by anybody with a web browser accessing the links form the MIBL site (http://www-ners.engin.umich.edu/labs/mibl/) only when the experiment are running and the Appletview server is turned on. Occasional miscommunications can occur mainly if there are networking problems at the server's site, but they can easily be fixed by shutting down and re-starting the Appletview server.

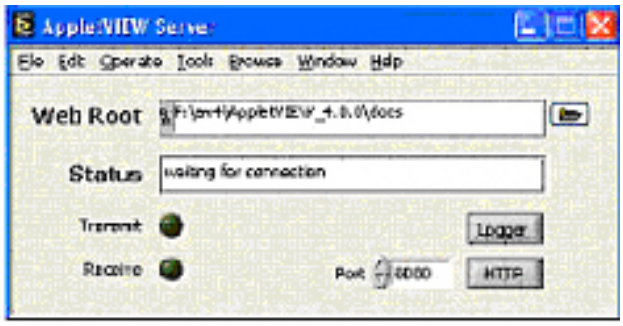

Figure1. Server VI

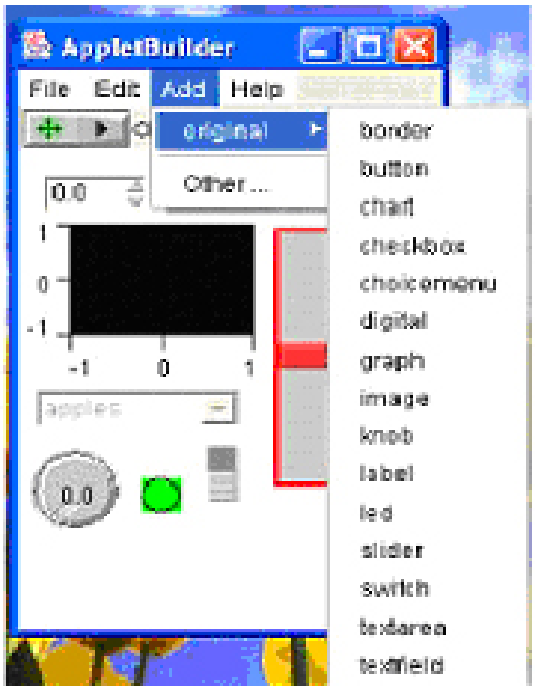

Figure 2. AppletBuilder screen shot

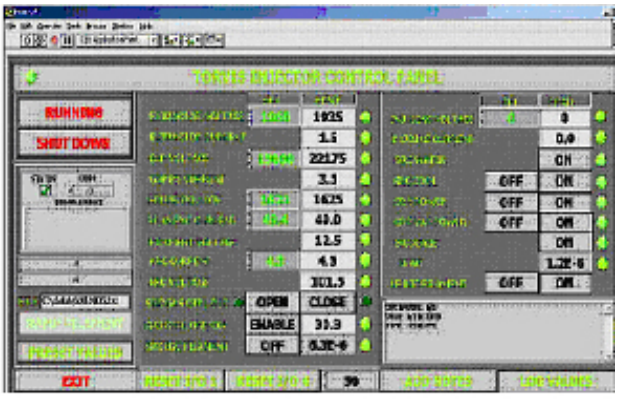

Figure 3. Server interface

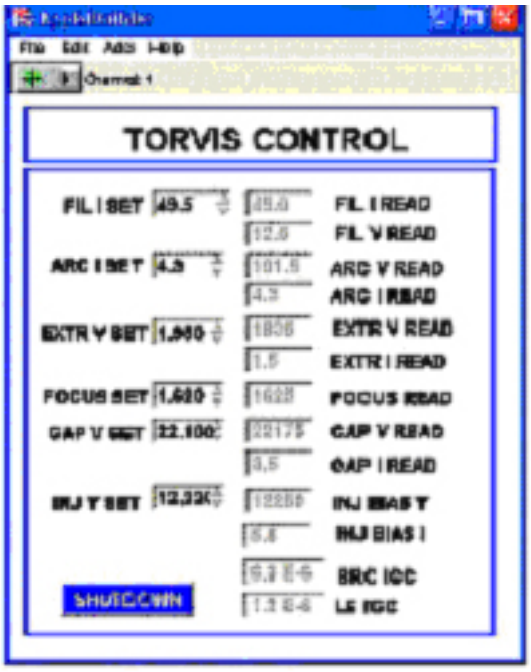

Figure 4. Client interface

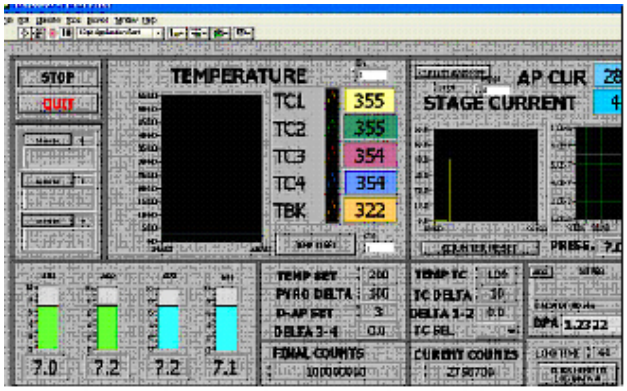

Figure 5. Server computer for the stage monitoring

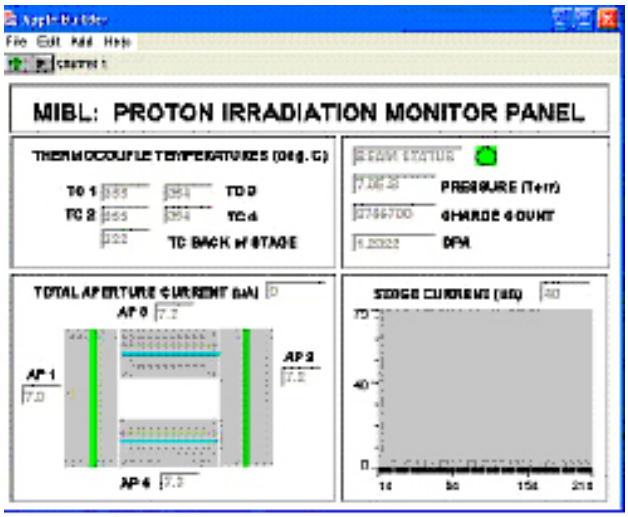

Figure 6. Client web view

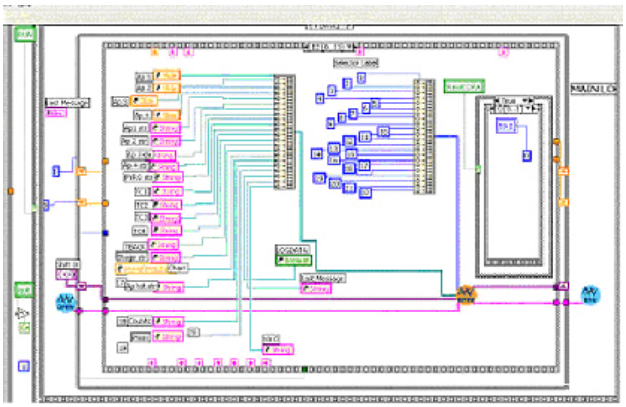

Figure 7. Back diagram of the Labview program on the stage computer

VII. CONCLUSION

At MIBL we were able to find a satisfactory solution to remotely monitor and control irradiation experiments. The solution with Labview and Appletview is relatively simple, not expensive and was developed using commercially available software packages and in-house programming. As a next step, we would like to improve the remote interface by adding more features and more control. Among other things, it is desirable to be able to remotely insulate parts of the beam line by shutting down the separation gate valves, to be able to reset the accelerator's power supply and to control the heating and cooling of the stage and of the samples. The initial assessment was made and all these could be implemented with a minimum amount of hardware purchase and with the use of Labview/Appletview combination

REFERENCES

[1] National Instruments: "Labview User Guide".
[2] Nancimiento Software Co.: "Appletview 4.0 Reference Manual"

AUTHOR

Ovidiu F. Toader, Research Specialist is with the Michigan Ion Beam Laboratory, University of Michigan Nuclear Engineering and Radiological Sciences Department, 2600 Draper Rd. Ann Arbor MI 48109, USA (e-mail: ovidiu@umich.edu).

Manuscript sent on October 05, 2005.