XVI IMEKO World Congress
Measurement - Supports Science - Improves Technology - Protects Environment ... and Provides Employment - Now and in the Future
Vienna, AUSTRIA, 2000, September 25-28

DESIGNING DISTRIBUTED DAQ SYSTEMS USING JAVA

P. Bobinski, R. Lukaszewski and W. Winiecki
Institute of Radioelectronics
Faculty of Electronics and Information Technologies
Warsaw University of Technology, 00-665 Warsaw, Poland

Abstract: A new concept of distributed data acquisition systems using Java architecture is proposed. A connection of external modules (standard library for National Instruments DAQ board) to source code in Java is described. Next, an example of the system, that uses a communication based on sockets for remote monitoring of meteorological parameters, also via Internet, is presented.
Keywords: Java, Virtual Instruments, DAQ

1 INTRODUCTION

Development of information technologies leads up to appear advanced network services, such as publishing various information (WWW) or remote operating of application. One of the possibilities of applying new networks technologies in metrology is remote access to advanced metrological services, including specialised processing procedures, unique measuring instruments, access to novel measuring systems and to research laboratories. Using distributed networks enables one to simultaneous applying above elements by many users independently from their place of stay. In the paper a concept of virtual instruments system is presented, as a part of Virtual Laboratory in Internet, that is realised as a Dean's grant. Main deal of the described work is to ensure remote controlling and data collecting from data acquisition (DAQ) board, both via distributed network, in particular: designing program interfaces and dialogue interfaces for virtual meteorological station.

2 THE NEW CONCEPT

There is many methods to ensure the user remote access to DAQ [6] from terminal that is connected to LAN/WAN network. All these methods are based on architecture Client-Server using standard network protocols (for example: TCP, IPX, UDP). Most of the solutions [2], [3], [4], [5] is based on integrated environments as LabView, LabWindows or HPVEE, that have built in procedures that ensure an access to network transmission protocols. Easy and fast designing of complex graphical user interfaces is a great advantage of these solutions. Their basic disadvantage is a fact, that client program, created in this way, is a separate application and must be sent to user computer, installed there and run on.

There is a possibility to use WWW mechanism that enables one to remote control a measuring system from WWW page using standard Internet browsers (as: Netscape Navigator, Internet Explorer). Such an access to available services is very easy for users, so we decided to apply Java architecture in the project and we have taken an assumption, that remote control application will be accessible as an applet via WWW server. Data flow diagram is presented in Figure 1.

3 IMPLEMENTATION

There were taken the following assumptions to realise the software:

- an mediator role between measurement application (applet) and DAQ function library is held by TCP/IP server that interprets (coming on an appropriate communication port) commands and calls board driver functions; this server (called GPIB/DAQ server) is an extension version of GPIB server, presented in [1],
- a Java class for communication between an applet and the server is created,
- a special protocol (described in [1]) for communication between the GPIB/DAQ server and an applet (strictly: a class, that is used by the applet) is applied,
- the applet with the above class have to be available from WWW page using HTTP protocol.

Designed software was divided into four functional modules: data source interface, data source server, client interface, server WWW. As a data source (Fig. 2) are measurement sensors connected to DAQ board. Data source interface consists of DAQ board (National Instruments) and also software drivers to controller board and libraries ("nidaq32.lib") supported by producers.

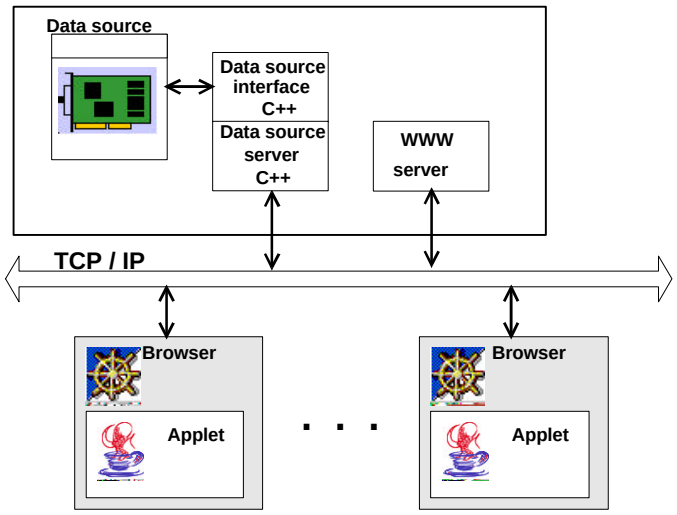

Figure 1. Data flow diagram in Virtual Laboratory.

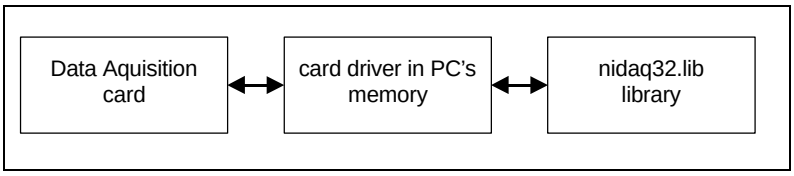

Figure 2. Block diagram of data source interface.

Data source server (called as DAQ server), created in Visual C++ 5.0, is based on data transmission via sockets (TCP/IP protocol) [7], [8] and includes an interpreter of commands, that are coming on appropriate communication port. It takes a function of an mediator between measurement application (applet) and DAQ function library. Block diagram of socket based conception of controlling DAQ card via WAN is shown in Fig. 3.

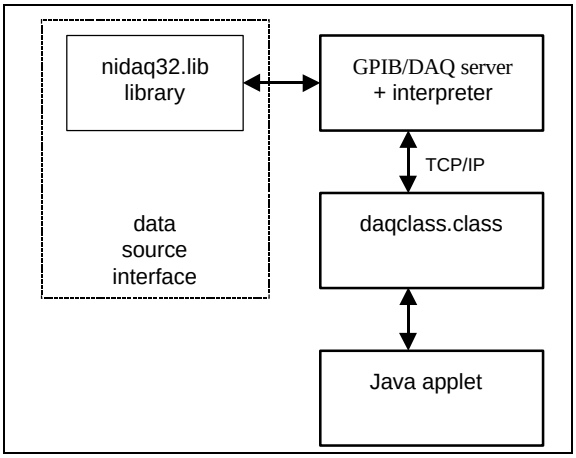

Figure 3. Block diagram of socket based conception of controlling DAQ card via WAN.

Source data interface proposed in [1] was developed by adding DAQ board control module to the GPIB server. It takes a function of an mediator between measurement application (applet) and DAQ-function library.

Construction of source data server, Java class and their common communication are similar to solutions proposed in [1]. Some new elements are as follows:

– connection of nidaq32.lib library to the server,
– development of the interpreter (included to the server) for DAQ board functions,
– the class daqclass.class covers methods, that are appropriate to the DAQ board functions.

A realisation of the function, that performs a single measurement of voltage in chosen input channel, is presented below: from the server (interpreter) point of view - example 1, and from the Java class (daqclass.class) - example 2.

```
switch(fun)
…
case 71:
        podziel_na_pola();
        sprintf(str,"AI_VRead(%i,%i,%i,&dVoltage)",atoi(Pola[0]),atoi(Pola[1]),atoi(Pola[2]));
 LogToFile(str);
 iStatus=AI_VRead(atoi(Pola[0]),atoi(Pola[1]),atoi(Pola[2]), &dVoltage);
        if (iStatus != 0)
         sprintf(str,"99(%s)", "Error DAQ NI");
        else
         sprintf(str,"71(%f)",dVoltage);
 break;
```

Example 1. Function (from server side) for voltage measure on DAQ's analog input.

```
   public double daq_ai_read (int device, int chan, int gain)
   {
        int i = 0;
        double r=0;
        str = "";
        String[] strarray = new String[1];
        out.println("71("+device+";"+chan+";"+gain+")");
        try { str = in.readLine();
                if (str.startsWith("71")) {
                        System.out.println("OK " + str);
                        getfields(str,strarray);
                        try{ r=Double.valueOf(strarray[0]).doubleValue();
                          } catch (NumberFormatException e) {return -2;}
                }
                else System.out.println("Err "+ str);
            } catch (IOException e) {
              System.err.println("Couldn't get I/O for the connection.");
              return -1;        }
   return r;
   }
```

Example 2. Method (in daqclass.class) for voltage measure on DAQ's analog input.

To build a client interface (Java application or applet) specialised class (DAQ.class) was created, that contains methods corresponding to functions of DAQ board.

Communication between Java applets (or application) and DAQ server is carried out using protocol specially designed for this purpose. This protocol was created to limit transmitted data amount, to limit a possibility of faults appearance when a measuring system is working, and to easy detection and interpretation of errors.

Simple WWW server was created in Java to make available (HTTP protocol) client applet with DAQ.class.

4 EXAMPLES

Presented concept was a starting point for building the measurement applications. A monitoring station was designed. A set of sensors was placed on printed circuit board, and connected to DAQ board. The board was plugged into the PC-type computer. The station enables one to measure a temperature, humidity and lightness. The applications, accessible using standard browsers, were built as the applets. The applet uses directly DAQ.class methods. Panel of virtual meteorological station was designed. The panel covers:

- push buttons *Connect* and *Quit*, responsible for communication connection and disconnection with the server (connection with the station automatically run its initiation),
- a set of dialogue fields, that enables one to set basic parameters of data acquisition,
- push buttons *Go*, that enables one to send measurement parameters to the DAQ board and to initiate a measurement procedure,
- graphic fields for displaying voltage values from four analogue inputs of DAQ board.

Most of the graphic user interface objects generates events (in the moment of user action, for example: pushing a button), that are serviced by the software.

Simpler version of meteorological station panel is shown in Fig.4.

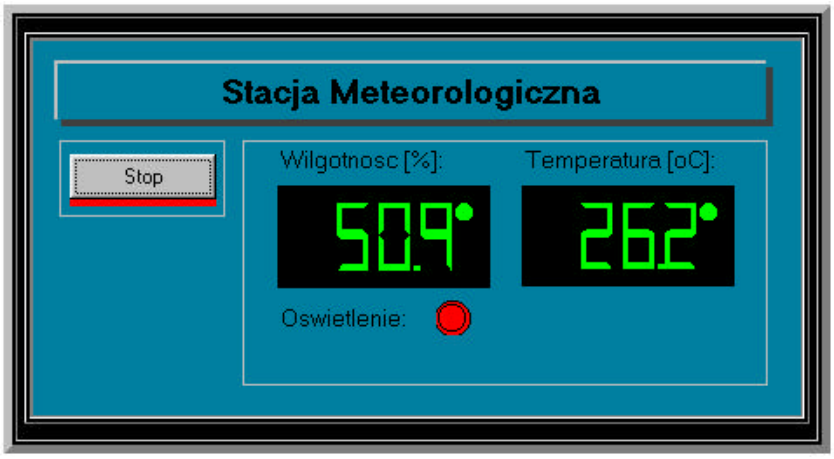

Figure 4. Panel of virtual meteorological system controlled via WAN.

5 CONCLUSIONS

As a result of presented research, a set of software tools was prepared, that enable us to realise virtual instruments with DAQ board and to control it's via distributed network. A set of virtual panels of meteorological station, that is a part of Virtual Laboratory in Internet was worked out. A kind of collected measuring data depends only on a set of measurement sensors connected to a National Instrument data acquisition card. This example has confirmed usefulness of proposed concept and wide possibilities of development of distributed data acquisition systems based on plug-in boards.

Application of function library (API) of DAQ board let us to design, in a simple way, the data source interface, and to simple realisation of software interface. The software, that was designed during the project, enables one to use almost all National Instruments DAQ plug-in boards to design distributed virtual instruments. The software interface enables us to control, for example, the following boards: PXI-1010 (PXI Chassis with SCXI slots), SCXI-1126, PC-6503, PCI-6110E, PCI-6111E, DAQPad-6507 (USB-DIO-96) simplify.

REFERENCES

[1] R. £ukaszewski P. Bobiński, W. Winiecki Java-Based Distributed Ieee-488 Measuring Systems, in: *Proc. IMEKO 2000 World Congres* (Vien, September 2000)
[2] L. Benetazzo, M. Bertocco, F. Ferraris, A. Ferrero, C. Offelli, M. Parvis, V. Piuri, A Web-Based Distributed Virtual Educational Lab, in: *Proc. 16th IEEE IMTC Conf*. (Venice, May 24-26, 1999), CD.
[3] G. Fortino, L. Nigro, A Multimedia Networking-Based Approach to the Development of Distributed Virtual Instruments, in: *Proc. 16th IEEE IMTC Conf.* (Venice, May 24-26, 1999), CD ROM.
[4] M. Bertocco, M. Parvis, An Auto-Routing Multi-Server Architecture for High-Education Training on Instrumentation and Measurement, in: *Proc. 16th IEEE IMTC Conf.* (Venice, May 24-26, 1999), CD.
[5] M. Bertocco, F. Ferraris, M. Parvis, A Remote And Distributed Student Laboratory, in: *Proc. IMEKO XV World Congres* (Osaka, June 13-18, 1999), 1999, CD ROM.
[6] NI-DAQ Function Reference Manual for Windows, NI Corp., Austin, 1996.
[7] http://www.sockets.com/
[8] http://www.sun.com/

AUTHORS: Ass. Prof Wies³aw WINIECKI, Ph.D., Ass. Piotr BOBINSKI, M.Sc. and Ass. Robert LUKASZEWSKI, M.Sc., Institute of Radioelectronics, Faculty of Electronics and Information Technologies, Warsaw University of Technology, 00-665 Warsaw, Poland, Phone / Fax: +48 22 825 52 48, E-mail: W.Winiecki@ire.pw.edu.pl