Journal of Cybernetics and Informatics

published by

Slovak Society for Cybernetics and Informatics

Special Issue
"New Trends in Education of Automation and Information Technology"
2004

http://www.sski.sk/casopis/index.php (home page)

ISSN: 1336-4774

AN OPEN REMOTE AND VIRTUAL LABORATORY FOR AUTOMATIC CONTROL TEACHING

Balda, P., Schlegel, M., Štětina, M.
University of West Bohemia in Pilsen, Department of Cybernetics, Univerzitní 8, 306 14 Plzeň, Czech Republic
pbalda@kky.zcu.cz, schlegel@kky.zcu.cz, milan.stetina@seznam.cz

Abstract: *This paper presents an approach to the building of remote and virtual laboratories based on industrial standards and utilizing the compatibility of control system REX and Matlab-Simulink. The approach is demonstrated on the PID autotuning laboratory configuration which has been developed and is running at University of West Bohemia in Pilsen, Czech Republic.*

Keywords: *Remote laboratory, PLC real-time control, PID autotuning, OLE for Process Control.*

1 INTRODUCTION

A remote access laboratory for undergraduates is currently being built at University of West Bohemia in Pilsen, Czech Republic.

There are many approaches to the building of virtual and remote laboratories, although the development is proceeding only for about a decade. The explosion of internet technology has been the most important motivating factor for these approaches. The key role is played by the internet communication between laboratory clients and remote laboratory devices (remote laboratory) or remote simulation engine (virtual laboratory).

An early but fundamental work (Aktan *et al.* 1996) solves the communication problem by stand-alone client-server application called "Second Best to Being There" where client and server parts communicate using UDP/IP protocol. The remote laboratory user interface consists of several hard-coded windows. Later applications incline to use standard web browsers for the laboratory user interface. Several main streams can be observed:

Scripting. Various client and server scripting technologies are used, e.g. CGI (Common Gateway Interface), (Parkin *et al.* 2000).

Java language. Java offers several technologies (JScript, Java applets, Java programs) which are well portable between different platforms (Windows, Linux). Complete Java-based solution is provided in (Röhrig and Jochheim 1999), (Jochheim and Röhrig 1999), and (Röhrig and Jochheim 2000).

ActiveX components. This Microsoft's technology is based on COM (Component Object Model) technology and it enables to insert user code components into web pages. This technique is for example used in Labview Virtual Instruments (National Instruments) or in WebHMI (Iconics).

Standard packages. There are several powerful commercial standard packages e.g. LabVIEW, Matlab-Simulink (and Real-time Workshop), etc. which has been equipped with web interfaces. These interfaces are usually implemented using the combinations of the techniques above.

The shift from very special dedicated application to the present state open industrial standard approaches and global program packages is observable in the field of remote and virtual laboratories development.

This paper describes such an approach to the building of the remote and virtual laboratory for control engineering undergraduate courses at University of West Bohemia, Faculty of Applied Sciences, Department of Cybernetics. Our solution is based on:

- Matlab-Simulink used for virtual laboratory simulation.
- Industrial control system REX for remote process model control.
- An arbitrary process station with REX runtime.
- An OPC (OLE for Process Control) compliant HMI (Human Machine Interface) package.

The main advantage of REX, (Balda *et al.* 2003), is its compatibility with Simulink. A particular laboratory implementation exploiting WinPLC (or PC) as the process control station and Genesis32/WebHMI as HMI, is shown at the end of this paper.

2 LABORATORY STRUCTURE

General scheme of a remote and virtual laboratory based on Web server, Simulink and REX is depicted in fig. 1. Laboratory components – process station, server, internet communication and client – are described in a more detail in the following subsections.

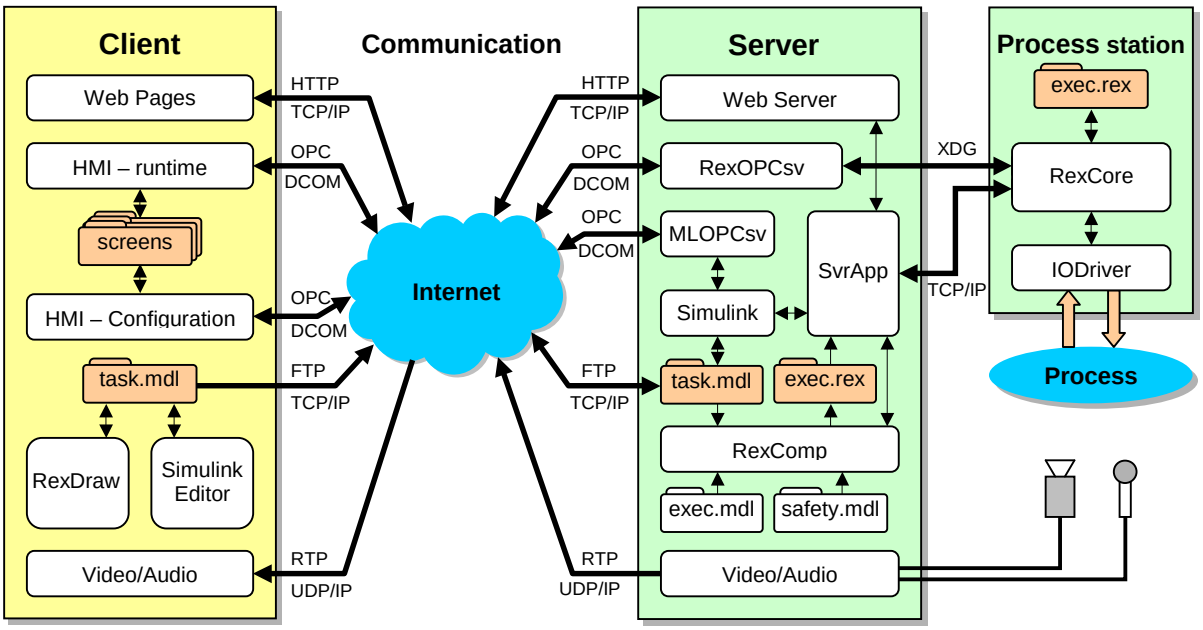

Fig. 1. Structure of remote and virtual laboratory based on system REX.

2.1 Process station

The *Process station* performs direct control action to the experimental device – process. The process station consists in a hardware platform, which supports runtime version of REX. At present, the following devices support REX runtime:

- WinPLC device running Windows CE operating system and equipped with I/O modules of the DL 205 PLC series (Koyo)
- Arbitrary PC or IPC (industrial PC) running Windows 95/98/ME/NT4/2000/XP or Phar Lap ETS and equipped with various plug-in I/O boards
- Arbitrary device equipped with Windows CE 4.x .NET

Program RexCore (real-time core of REX) is the most important software of the process station. RexCore calls an appropriate input/output driver module IODriver depending on the process interface hardware used. There are drivers for WinPLC modules, Advantech plug-in boards and modules, Ethernet Base Controller using Terminator I/O and/or DL 205 modules, Modbus master and slave devices and last but not least is the OPC driver which communicates with any device supporting OPC Data Access server of versions 1 or 2.

After the process station restart, the program RexCore is launched and its initial configuration is read from the file `exec.rex`. Communication between RexCore and the server is performed by XDG (reX DiaGnostic) protocol which is implemented above TCP/IP.

In the case of time noncritical (slow) process, it is not necessary that the process station is a stand-alone device, it can be a part of the server computer.

2.2 Server

The *Server* computer is a gateway to the laboratory. The server runs one of the Microsoft's server operating systems, e.g. Windows 2000 or XP Server.

Besides web pages providing, it serves both operating modes (virtual and remote laboratory) using the program `SvrApp` which enables:

- User login and operation mode selection.
- Selection of one of the pre-built example configuration or a user developed configuration.
- Start/Stop the experiment.

The chosen operating mode influences further server activities.

Virtual Laboratory

In the virtual lab mode, the program `SvrApp` launches Matlab-Simulink with the selected example or user developed configuration. The file `task.mdl` (fig. 1) corresponds to the experiment°configuration which contains both the simulation model and the control algorithm. Data from the experiment are gained and user commands to the experiment are sent through Matlab OPC server `MLOPCsv`.

Remote Laboratory

The same file `task.mdl` is a part of the configuration of REX. Two additional pre-configured files of REX configuration are displayed in fig. 1:

`safety.mdl` – second task in the REX configuration. It supervises the experiment on the remote device is safe. Its configuration depends on the particular experimental device character and it enables to stop the experiment if some measured value leaves the permissible limits or it has unacceptable dynamic behavior (big oscillations, etc.).

`exec.mdl` – the REX project main file. It includes timing properties of the tasks `task` and `safety` and configuration parameters of the driver `IODriver`.

After the user selection or creation of the file `task.mdl`, the program `SvrApp` calls the REX configuration compiler `RexComp` which compiles the project main file `exec.mdl` and generates the file `exec.rex`. If any error occurs during the `task.mdl` compilation, the user obtains the error list in the form of a web page, otherwise the compiled configuration `exec.rex` is sent to the executive `RexCore` by means of the download command of XDG protocol. The downloaded configuration is started immediately on the target device. The experiment running can be watched and manipulated through the OPC server `RexOPCsv` of the system REX.

Video and audio signals are transmitted live from the special board in the server computer, camcorder and microphone.

The compatibility of REX and Simulink allows the user to design the file `task.mdl` so that it can be used for both the simulation in Simulink and the real-time control in REX without any change.

2.3 Internet communication

Several standard protocols are used for data transmission between server and clients:

`HTTP` (HyperText Transfer Protocol) is used for transmission of web pages from server to clients and for sending the filled form data to the server.

`FTP` (File Transfer Protocol) is used for sending the user configuration files to the server (e.g. `task.mdl`).

`OPC` (OLE for Process Control) is used for periodical data reading from the experiment, for the specification of experiment modes and parameters and also for the browsing of signals which are available in the OPC servers.

`RTP` (Real-Time Protocol) is used for audio and video signals transmission.

The first two protocols are used only from time to time on the user command, the latter pair is used periodically during the experiment run.

2.4 Client

Finally, the software of a client (laboratory user) computer which is connected to the laboratory at least by a modem (or ISDN modem) is described. Client computer is equipped with some of the Windows operating systems (the best are Windows 2000 or XP) and Internet Explorer at least version 5.

Human machine interface (HMI) is performed with some standard OPC DA (at least version 2) client tools, for example Genesis32 by Iconics, InTouch by Wonderware, Control Web by Moravian Instruments, Promotic by Microsys and many more. These program packages usually provides tools and components for remote visualization and manipulation using Internet. After the automatic download and installation of the tools the user can immediately use all pre-built configurations of the virtual and remote laboratory.

For the development of user's own algorithms two requirements have to be allowed:

1. Configuration of user own control algorithms,
2. Creation of user own HMI screens.

In our concept the first requirement means that the user is allowed to create his/her own files with the `.mdl` extension. If the user have got a Matlab-Simulink licence, he/she can use internal Simulink graphic editor, else the program RexDraw which is a part of REX should be used.

The second requirement can be fulfilled with the demonstration versions of some HMI or SCADA systems which are free of charge. These demoversions have usually some restrictions concerning maximum number of I/O points but it need not be a problem in cases of simple experiments.

Audio and Video signals from the remote laboratory can be integrated directly to the client HMI screens.

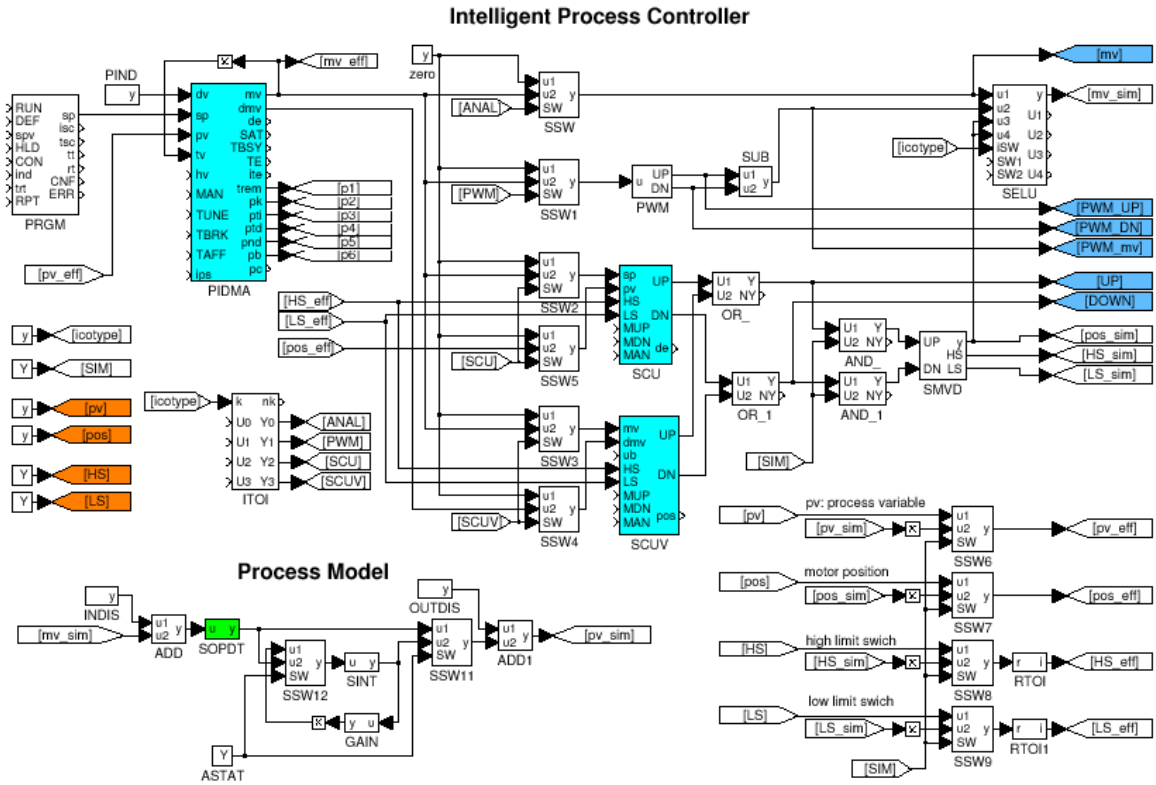

Fig. 2. Configuration of a PID autotuning laboratory for Simulink and REX.

3 PID AUTOTUNING LABORATORY

This section presents the first remote and virtual laboratory application at University of West Bohemia in Pilsen, Czech Republic.

Fig. 2. depicts the configuration of the virtual laboratory for the automatic tuning of PID controllers. The configuration consists of the universal PID autotuning control loop and the process simulation (bottom left part of the figure). The function block `PIDMA` which implements PI(D) controller with Moment Autotuner, (Schlegel *et al.* 2002), is the heart of the configuration. Control loop supports various industrial controller output types, which are user selectable:

`ANAL` – Control loop with the conventional analog output,

`PWM` – Pulse Width Modulation output,

`SCU` – Step Controller Unit output with position feedback,

`SCUV` – Step Controller Unit with Velocity input signal (without position feedback).

The last figure 3 shows the example of HMI screen for this control loop application.

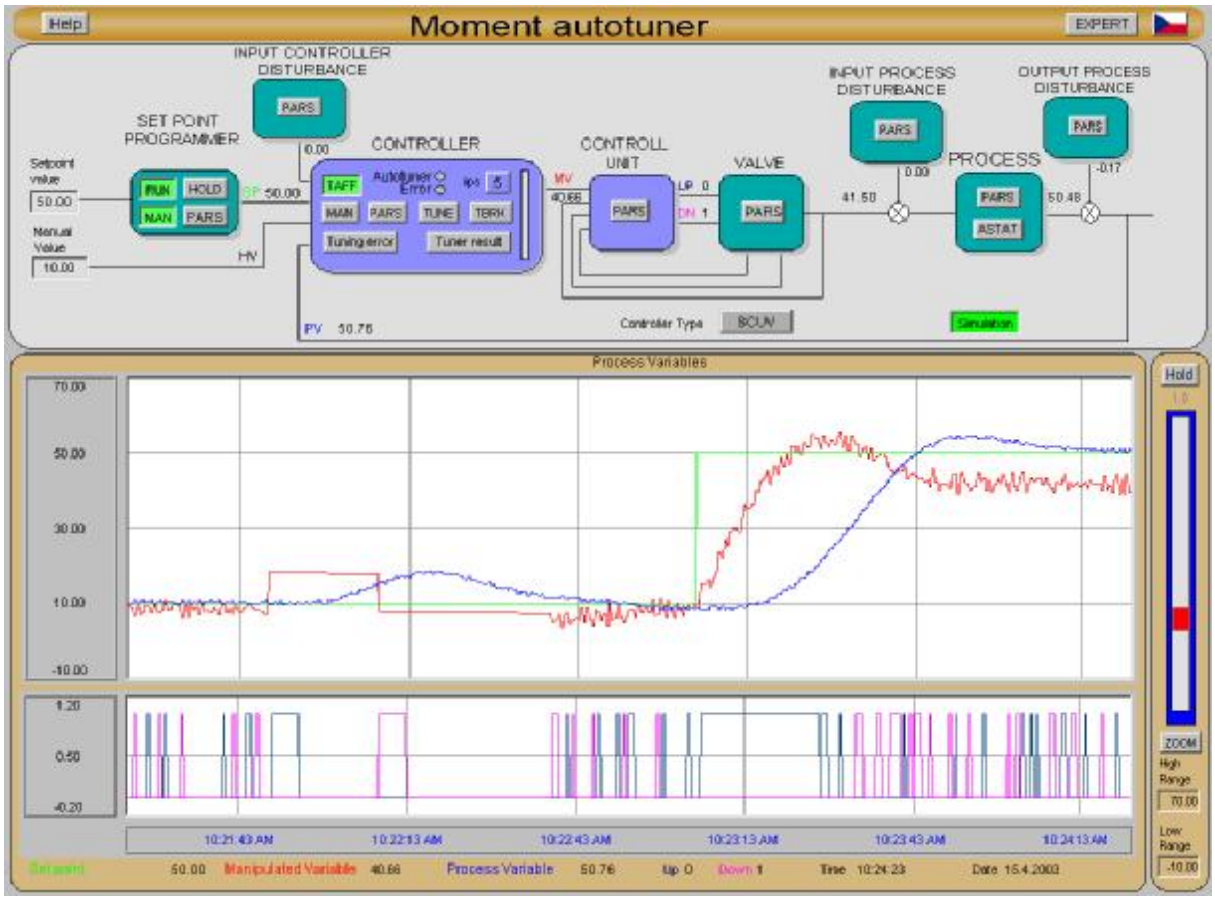

Fig. 3. Example of PID autotuner loop in Genesis32 and WebHMI.

4 ACKNOWLEDGMENTS

This work has been particularly supported by MŠMT ČR - project no. MSM 235200004 and GAČR - project no. 102/02/0425. This support is very gratefully acknowledged.

5 REFERENCES

Aktan, B., Bohus, C.A., Crowl, L.A. Shor, M.H. (1996). Distance learning applied to control engineering laboratories. *IEEE Transactions on Education,* Vol. 39, No. 3, 320-326.

Balda, P., Schlegel, M., Štětina, M. (2003). New control system REX supports design and simulation in Matlab-Simulink. *Automatizace*, Vol. 46, 2/2003, 93-96 (In Czech).

Iconics, Inc.: http://www.iconics.com.

Jochheim, A., Röhrig, C. (1999). The virtual lab for teleoperated control of real experiments. In: *Proceedings of the 38th IEEE Conference on Decision and Control*, Phoenix, USA, 1, pp. 819-824.

National Instruments: http://www.ni.com.

Parkin, R.M., Czarnecki, C.A., Safaric, R., Calkin, D.W. (2000). A PID servo control system experiment conducted remotely via Internet,. *Mechatronics*, Vol.12, 833-843.

Röhrig, C., Jochheim, A. (1999). The virtual lab for controlling real experiments via Internet. In: *Proceedings of the 11th IEEE International Symposium on Computer-Aided Control System Design*, Hawaii, USA, 279-284.

Röhrig, C., Jochheim, A. (2000). Java-based framework for remote access to laboratory experiments. In: *IFAC/IEEE Symposium on Advances in Control Education*, Gold Coast, Australia.

Schlegel, M., Balda, P., Štětina, M. (2002). Robust PID-autotuner: Method of moments. In.: *Proceedings of the conference Process Control 2002*, Kouty nad Desnou, Czech Republic, June 9-12 (In Czech).