INTERNET-BASED CONTROL ENGINEERING LABORATORY

Tudor Buzdugan*, Robin De Keyser, Ioan Nascu***

*Technical University of Cluj–Napoca
Faculty of Automatic Control and Computer Science
Tudor.Buzdugan@personal.ro, Ioan.Nascu@utcluj.ro

**EeSA – Department of Electrical energy, Systems and Automation
Ghent University
rdk@autoctrl.Ugent.be

Abstract: The motivation, structure, and performances of an educational Internet-based control engineering laboratory are presented. It is intended to be a general tool, able to work with a large category of real processes with minimum constraints, and applicable to all kinds of control algorithms that are usually taught in school. It will be shown the ease of using such a system from both, teacher and student, points of view.

Key words: distance learning, laboratory experiments, real-time, control engineering, internet.

1. INTRODUCTION

Often, in control engineering schools, the real life experiments are replaced by computer simulations, this being a cheap alternative to providing laboratories with support for a large number of students. Unfortunately, simulations cannot provide an accurate description of the real process. This is a big weakness of the simulations and for this reason the results that were obtained in simulations have to be tested on the real process in order to verify their accuracy. In fact, it is highly probable that in a real experiment, the results will differ from the ones obtained via simulation.

The purpose of this project is to make teachers and students rediscover the laboratory experiments, with less resources and lower costs. For this reason, a complete system, able to connect the student workstation with the real plant has been developed. The student can now perform laboratory experiments in the lab or at home, via internet, without too many constraints.

Some other attempts in the field of online laboratories have been analyzed. The general conclusion was that they are either simulations [1], either designed for a set of plants and therefore not a general solution [2], either rely on expensive software [3].

This system is intended to be generally applicable, for a large category of real processes and to provide support for the most common and specific laboratory experiments in the field of control engineering. One of the most important aspects is the feedback that the student gets from the experiment. All the data collected from the plant will be made available for the student in both graphic and brute forms.

Because even if working with a real plant the student will not get the hands-on experience that would be obtained when being right next to the process itself. Visual, audio and any other kind of information that will provide the "feeling" of the plant can be made available, but these are left for the teacher to take care about, him being the one that knows the particular experiment he intends to use. This feedback can easily be integrated in the system although being impossible to give a general solution.

The project was developed under GPL (General Public License) and a wide variety of technologies, starting with the user interface graphics and going deep into the kernel of the operating system, was integrated making possible a total transparency between the student and the real plant at the lowest possible cost.

2. ASPECTS ON THE IMPLEMENTATION OF THE SYSTEM

In order to be a general system, the following requirements have been imposed:

- to have a user (student) interface that can run on any platform;
- to be able to work with a general type of real processes;
- to be able to work with multiple plants;
- to allow easy custom set-ups and extensions.

Therefore, it has been chosen the system structure presented in Figure 1. It consists in an embedded web server, able to run real time tasks, along with some interfacing hardware to the real plant.

On the student side, a simple PC, connected to internet and a web browser is necessary, as all data collected from the plant is given back to the student and he can use other software tools to process it. Anyhow, a plot of these data is provided for fast experiments, and on-line plots are available for slower processes that allow real-time data exchange over the internet.

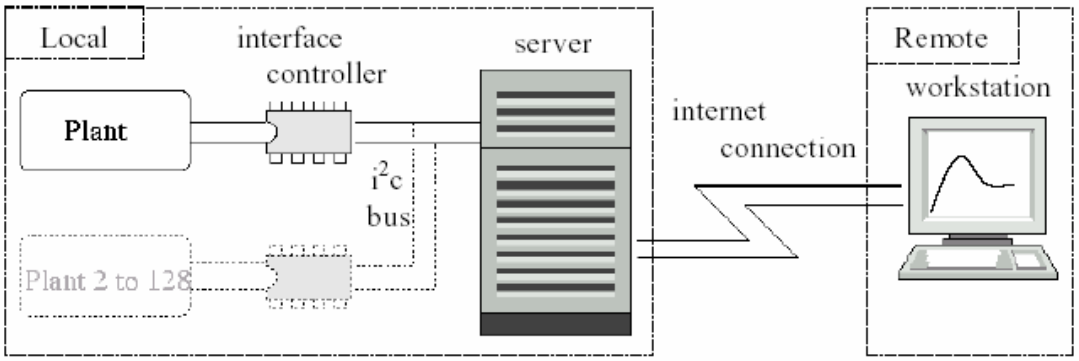

Figure 1. Structure of the system.

The "Local" part of the system will be discussed in the following. First will be given the description of the general plant that the system can handle.

2.1. The Plant

In order to provide generality, the system can handle experiments with a maximum of two inputs and two outputs, allowing the use of multivariable plants or load/disturbance control for Singe Input - Single Output systems.

The laboratory experiment to be connected to the system is left for the teacher to care about. It can be any plant with a maximum of two analog outputs and that can take PWM control signals as inputs.

During system development a DC motor speed control, with controllable load, has been used. The schematic picture of such a possible experiment is presented in Figure 2.

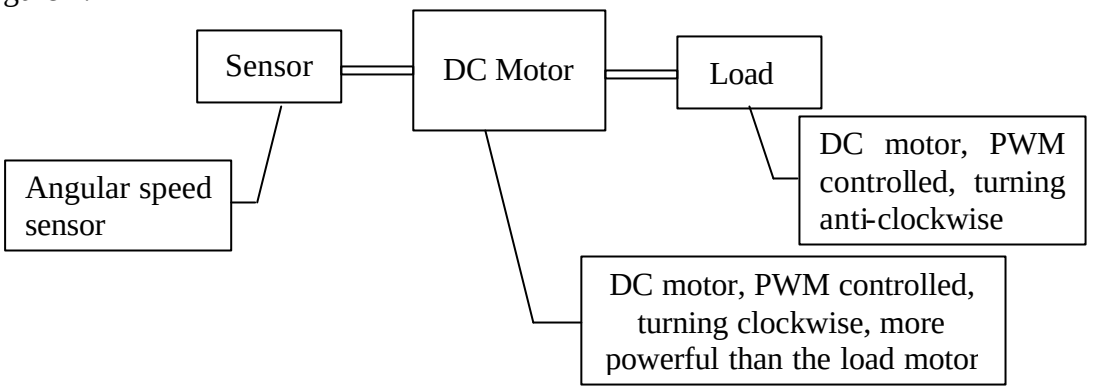

Figure 2. Schematic of a possible laboratory plant.

The time constants of this system are of the order of 0.3 seconds and the system behaved well. Fast processes are recommended because the time needed to perform experiments becomes very short. For this plant, experiments of one minute are more than relevant.

When choosing the plant to work with, the teacher should keep in mind that the minimum sampling time is of 10 milliseconds.

2.2. The Server

The main part of the system is the server that assures the connectivity between the student's workstation and the plant. It has two functionalities:

- to provide the internet web service and
- to provide the real-time environment for controlling a real experiment.

In order to provide these functionalities the following GPL software was used: the Apache web server, to provide the service, along with the RTAI (Real Time Application Interface) patch and library, for the real-time environment, working together on a Linux platform.

The user interface consists of a web site, that the student sees in his browser, realized with PHP and JAVA SCRIPT technologies. Notice that each student has the possibility of performing his own experiments, therefore the web-page content must be generated on-line since it changes dynamically. PHP is one of the newest technologies for generating dynamic web pages, designed to work in Linux environment and therefore has been chosen for this project. JAVA SCRIPT technologies have been used for providing nice, user friendly, pages. This web site also contains documentation and a set of manuals for the student so the teacher will not need to spend time on teaching students how to use this tool.

The Apache web-server has been chosen to provide the dynamic web-page and full interactivity between the student and the real-time environment.

The real-time environment is provided by RTAI, a real time operating system based on the Linux kernel. The standard Linux kernel is not pre-emptable, meaning that the system is "locked" while a kernel function is executing, and these results in introducing nondeterministic latencies, not tolerable in a real time environment. RTAI is based on the HAL (Hardware Abstraction Layer), adding a new software layer beneath the Linux kernel with full control of interrupts and processor key features. The real-

time Linux scheduler treats the kernel as a background task that is running when no real time activity occurs. In this environment, the Linux kernel task can never block interrupts or prevent itself from being preempted [5].

From the control engineering point of view the most critical real time activity is sampling the measured signals at constant rate. For this reason, data transfer activities between the server and the plant (via the interface controller) have been placed in a hard real time process in order to provide precise sampling times [4].

The control algorithms are currently built in the system and only their parameters can be varied. As the time needed to run these algorithms has been evaluated, they are running in hard real time as they are considered to be critical. Non critical routines (such as experiments start and configuration, collection of experiment data for the student) are run in soft real time.

In order to provide both services (web server and controller for the laboratory experiment), a trade-off between fast sampling and giving enough time to the web service to handle request had to be done. The system runs now on a Pentium 2 at 233 MHz PC and the critical sampling time, when there is no more time to run anything else but the data acquisition routine, is of 5 milliseconds. A minimum sampling time has therefore been chosen as 10 milliseconds. As this minimum of 5 milliseconds is due to conversion times and data transfers, it will not improve on a more powerful system, unless a faster I^2C interface is used (see The I^2C bus). The only benefit of having more computational resources is improving the time needed to run the control algorithms.

Having the described real time system running, the server is ready to send and receive data to and from the laboratory experiment.

2.3. The I^2C bus

The control signals provided by the control algorithm have to be passed to the real plant. Of course these signals are digital and more likely, the plant works with analog signals. The conversion is handled by a microcontroller and the connection between the server and the controller is done though a I^2C bus.

This type of bus has been chosen because nowadays it is very common for microcontrollers to have I^2C support built in, so no other hardware interfacing is needed.

A second reason for choosing I^2C is that multiple devices can be connected together and this allows connecting together multiple experiments, and the change between working with one plant or another simply means addressing the corresponding interface controller. This can be done from software and offers the student the possibility to work, at any time, with any of the experiments connected to the server with no hardware changes needed. This connectivity could not be provided by serial protocols and industrial bus protocols are not commonly built in controllers, additional hardware interfaces being needed.

An adaptor for I2C bus is needed for the server. A parallel port to I2C adaptor has been chosen because of its simplicity.

2.4. The Interface Controller

The main role of the interface controller is to convert digital data into analog signals and vice-versa. The control inputs to the plant, provided by the server in a digital form, are converted into PWM signals and analog outputs of the plant will be converted by the ADCs of the controller and forwarded to the plant. Multiple plants can be connected to the I^2C bus, each of them needing an interface controller. In order to differentiate between two plants, each controller will be given a unique address. As the I^2C pro-

tocol specifies a maximum of 128 different possible addresses, this is also the maximum number of plants that can be connected to the bus.

The controller used is a PIC 16x series, having two 8 bit PWM outputs and two 8 bit A/D converters. The digital control signals received from the server will be sent to the plant through the PWM outputs. The analog signals acquired from the plant have to be in the range of [0,5]V. The controller performs the conversions as fast as it can (each analog-to-digital conversion process takes approximately 14μs) and puts the results on the I^2C bus when requested. The same for the PWM output, the filling factor of the control signal is kept constant until a new value comes from the server. In this way, one does not need to care about synchronization between the server and controller.

3. BUILT IN EXPERIMENTS

For the moment, three categories of experiments are supported, covering some of the aspects of control engineering.

The first set of experiments is experimental identification of systems. It has been considered one of the most important experiments, in order to familiarize the student with the process he is going to control as well as for laboratory experiments dedicated to Identification of Systems discipline.

The step response and staircase experiments have been considered mandatory experiments in order to get the student familiarized with the plant. Results of such an experiment, performed on the plant previously described are presented in Figure 3.

Having so many identification methods available, the student is given free choice of the identification input signal, along with sample time and total experiment time settings. The student is able to feed any input signal, in the form of a file containing consecutive input values.

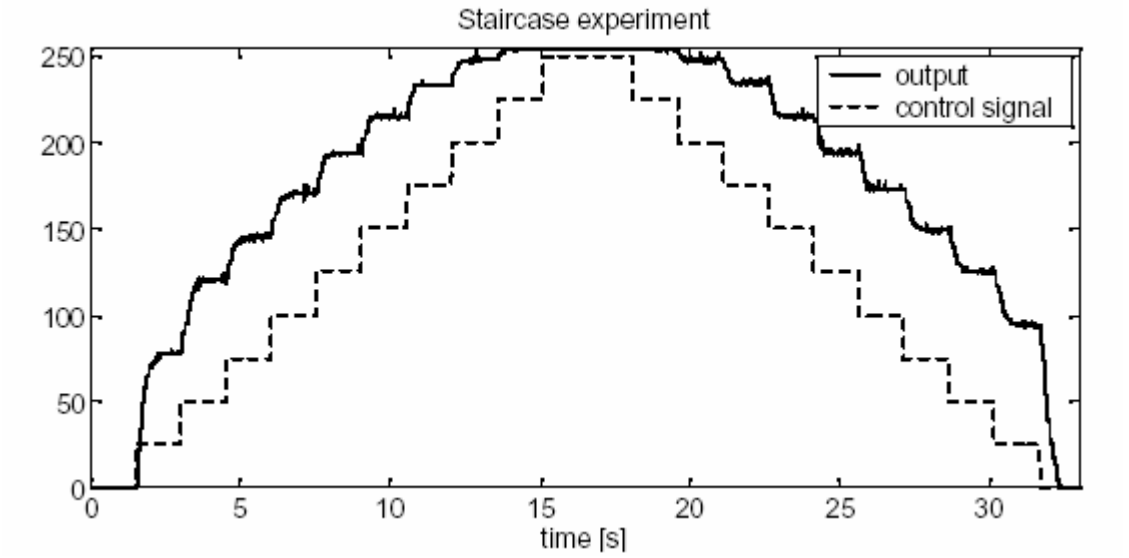

Figure 3. Staircase Experiment.

The second set of experiments consists of a PID controller. Again, time and controller settings have to be made by the student, input and output signals are plotted and a data file is made available.

Supplementary, reference settings can be made. In Figure 4 are presented results obtained for the same plant, with reference and load change.

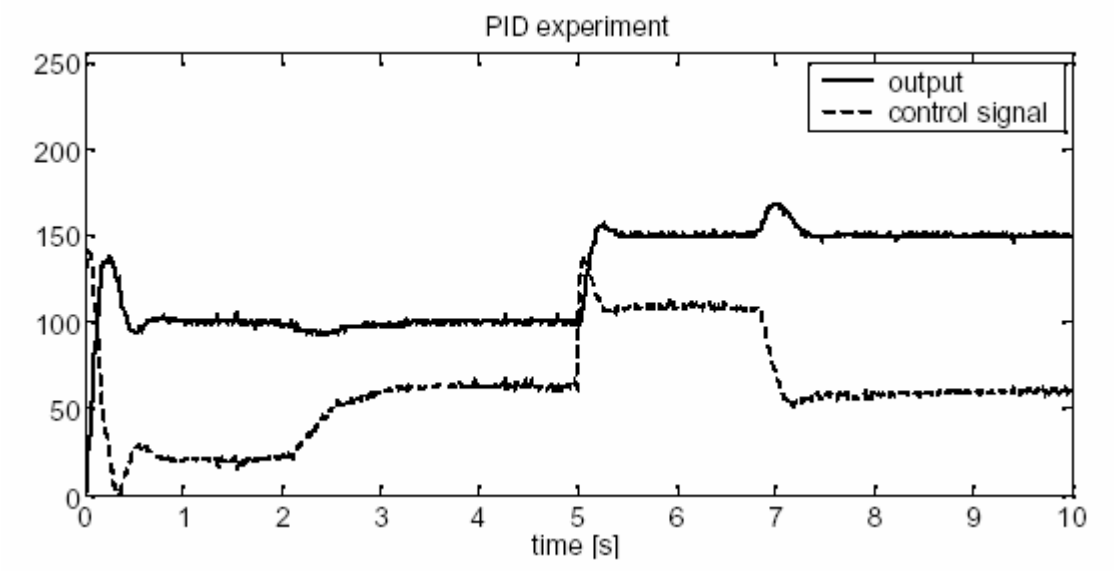

Figure 4. PID Experiment with reference and load change.

In order to make the student "feel" and understand how plant signals are transformed into control signals by the controller, a fuzzy controller experiment is the third built in experiment. The fuzzy control makes use of linguistic terms for describing signal values and semantic rules to generate the output, being therefore closer to human language than mathematical formulas.

The user interface is more complicated, the student having to set scaling values, number of fuzzy sets for each signal, times and the set of rules to compute the control signal. The fuzzy controller takes as input signals the error and its derivative. These signals are computed numerically.

Normalization, fuzzification, defuzzification and inference are performed by the system, the student providing adequate parameters for each routine.

4. CONCLUSIONS AND FUTURE WORKS

The implemented system, that is working as described above, is generally applicable to a large category of real processes and provides support for a set of specific laboratory experiments in the field of control engineering, at low costs and with low resources.

In order to be a more general tool, the system should be able to run any kind of control algorithm provided by the student in a high level programming language, specific to control engineering.

5. REFERENCES

1. http://www.esr.ruhr-uni-bochum.de/VCLab/
2. http://chem.engr.utc.edu/
3. B.C.Seet and K.V.Ling, Internet-Based Laboratory for Control Engineering Education.
4. Alan Burns and Andy Wellings, (2001), *Real-time systems and Programming Languages*, Addison-Wesley.
5. Herman Bruyninckx, *Real-Time and Embedded Guide*, *(http://people.mech.kuleuven.ac.be/~bruyninc/rthowto/rtHOWTO.pdf).*
6. http://www.oopic.com