Development of an XML-based lab for remote control experiments on a servo motor

R. Pastor, C. Martín, J. Sánchez and S. Dormido
Departamento de Informática y Automática, UNED, Madrid, Spain
E-mail: {rpastor, jsanchez, cmartin, sdormido}@dia.uned.es

Abstract In this paper, an XML-based framework, known as RELATED, is used to build a remote lab for providing experience of control engineering on a servo motor using components distributed across the Internet. This new approach allows different groups to share their hardware and software resources and to facilitate the creation of networks of virtual/remote laboratories.

Keywords distance learning; PID control; remote operations; virtual laboratories

Laboratory experiments play and will certainly continue to play an important role in control engineering education.[1] One significant tendency in the field of control practice is the increasing use of virtual instrumentation. In fact in major production facilities, operators are often trained in plant operation using a simulation environment (instead of the real process) to drive virtual instruments. What is obvious is that experimentation *in situ* with a plant or real object cannot be replaced by a simulation or training simulators, especially the sensations perceived by the student in the experiment.[2,3] Practical education needs to be based on errors and irregularities, as occurs in mechanical, electrical or chemical systems, as opposed to the ideal icons and environments represented on a computer display.

One vital aspect of control engineering education is therefore that laboratory and practical work need to provide engineering students with a taste of real situations, measurement and instrumentation, with all their attendant problems.[4] One idea is to allow the students to perform real experiments, in real time, on real equipment, but over the Internet.

The use of remote labs for supporting and integrating the activity of a control engineering course is in fact a widely discussed issue. An analysis of the recent solutions developed for remote labs, where different laboratory experiments are run remotely via a Web interface, is reported in Ref. [5]. Different solutions of remote control engineering laboratories are presented and discussed in Refs [6] and [7] where remote control was applied to a robot arm. A remote measurement lab is also described in Ref. [8]. In all the proposed solutions the remote user (the client) is connected via Internet to a dedicated Web server that interacts with the computers of the laboratory used for controlling (monitoring) the real processes.

The rapid progress of Internet technology and its increasing popularity has prompted several educational institutions to develop Internet laboratories.[9–14] With the help of an on-line Internet laboratory for control experiments, educators can be encouraged to design control engineering courses that combine theoretical issues and practical activities. Via the Internet, on-line laboratories could offer more flexibility to prepare assignments for students that require experimentation with real phenomena. In addition, an Internet laboratory allows a better use of equipment either by local or by remote users since they can access to the labs from anywhere and anytime with just an Internet connection. This sharing of resources not only brings down the experiment cost per student, but equipment will also be made available to more students since the time and space constraints normally associated with a traditional laboratory can be removed.

This concept of remote labs provides both vertical and horizontal integration of control education and addresses several problems faced by engineering educators such as the high cost of didactical setups, the duplication of resources when several departments attempt to offer their own control laboratories, the rapid obsolescence of equipment, and the difficulty of providing technical support.

In this general context, there are many works focused on the development of virtual and remote laboratories conducted through the Internet but all these remote and virtual laboratories are timely efforts by different research groups working independently.[10,13,15–19] Use of the software and hardware of other universities is not contemplated, and neither is taking advantage of work previously carried out by others. That is to say, until now, a methodology or a standard for the construction of networks of virtual/remote laboratories based on previous developments has not existed.

Therefore, formal specification of these laboratories becomes necessary, in such a way that the most expensive tasks can be automated; for instance, resource management or user administration. A separation is also obtained between the development of the laboratory (mainly when it is necessary to implement real-time control mechanisms), and the additional effort of ‘publishing on the Web’.

Laboratory specification

A formal specification for a virtual or remote laboratory must define transparent access to it, and this access must be independent of the type of system. To do this, it becomes necessary to use some linguistic mechanism that allows instructors to define these systems in an abstract way; that is, a meta-language is necessary. In this case, access to these systems (virtual or remote) will be made by means of an Internet navigator, therefore the idea of using the XML standard (Extended Markup Language) for the formal specification of these systems is obtained in a natural way. This XML-based specification declares a group of labels (components of the system) to describe the expected behaviour that will be implemented by means of the remote or virtual system (components include concepts like experiments and interactive visualisation).

Also, this specification must be implemented in a ‘run-able’ model. This model must allow ‘use’ of the laboratory and define the ways and methods of utilisation such as experiments, presentation of results, and access to hardware. This implementation must present the following characteristics:

- Access to the laboratory using ‘only’ a standard Java-enabled web browser.
- Reutilisation of the whole code generated for the ‘local’ use of the systems. In

case of remote systems, it is necessary to free users of the problems derived from the development of real time services.
- Access control to laboratories by using technologies of resource planning and user management.

Bearing in mind these characteristics, a Java implementation of the 'run-able' model has been developed allowing its utilisation from an Internet browser by using applet technology. This implementation – a set of Java classes implementing basic features – uses an XML specification – a Document Type Definition called DTD RLAB – that together represents the RELATED framework.[20,21]

RELATED basic XML definitions

XML is used in the definition of each RLAB System. All the objects included in an RLAB System are 'written' using an XML file in terms of tags and attributes. These elements formally represent the different laboratory elements that can be present in a virtual/remote setup. So an RLAB System is a set of different objects (explained below): variables, parameters, modules, experiments, views, and references. It is a black box that can be manipulated across the variables as happens in block-oriented applications (e.g., Simulink or LABVIEW). The ⟨**system**⟩ XML tag defines a system and **only** one tag is allowed on a definition file.

A **module** is an entity with variables and parameters, and it gives access to the user to get and set values. This entity runs code inside a local thread while a user is conducting a remote experiment. At the moment RELATED supports two types of implementation: Native code and Java code.

The development of module implementation is the biggest effort that an RLAB System administrator must do, but there are clearly many advantages. The ⟨**module**⟩ XML tag defines a module and the ⟨**implementation**⟩ tag is used to indicate what code will be run.

A **variable** is an object that stores a value (*double*, *float*, *integer*, *long*, *boolean*, or *string*) that can be changed while the system is running an experiment. A variable is always associated with a module since the module implements the **get** and **set** functions to access these variables. A variable is defined by the ⟨**var**⟩ XML tag.

A **parameter** is like a variable but it can not be changed while the system is running an experiment. Parameters are useful to initialise the internal variables of a module implementation.

A **view** defines a GUI form composed of graphical components and multimedia capabilities (video, animation, sound). These components let users visualise and manipulate the remote data of a RLAB System. The ⟨**view**⟩ XML tag defines a view and the syntax looks like a module definition.

A **reference** is a URL direction to publish information using HTML pages. Any information about local RLAB Systems can be added via references. The ⟨**reference**⟩ XML tag defines this object.

An **experiment** describes the dynamic behaviour of an RLAB System, such as, for example, a controlled system, manual operation, etc. An experiment is built by describing what modules are going to be used and how the module variables can set and get from other remote or local modules, and also if these variables can be changed by the student (i.e., they are interactive variables). Of course, the experiment duration must be indicated using the ⟨**duration**⟩ tag.

An RLAB System could include as many experiments as the system administrator adds to the definition file. Another possibility consists in allowing users to add their own experiments to the definition file.

The servo-motor laboratory

The 'local' laboratory used as an example is a didactical setup of a servo motor which has been developed by the Swiss Federal Institute of Technology in Lausanne (EPFL) team. Up to now, there are two approaches to teleoperate the servo-motor: a LABVIEW client and a Java-based client used in the eMersion project.[22] This last approach was integrated using the RELATED framework by reusing and refactoring the Java code developed,[23] but a new approach has been developed to provide new interfaces and features using RELATED.

The laboratory can be seen as a RELATED module that provides access to the real time data. So as a first step, the different variables and parameters (laboratory data) necessary for the remote conducting of experiments must be classified. In this example, the name of the variables for controlling the didactical setup are:

- SPEED. This represents the speed measurement.
- POSITION. This represents the position.
- CONTROL. This represents the controlled variable.
- KP, TI, and TD. Proportional gain, integral time, and derivative time of the built-in PID controller.
- SETPOINT. This is used in the closed-loop operation to set either the speed or the position reference.
- SPEED_OR_POSITION. Type of control.
- MANUAL_AUTOMATIC. Boolean variable to switch to automatic or manual mode.

In order to illustrate the features of RELATED in more detail, a case study is presented in which the steps to create a module and provide access to the real-time services of the laboratory are described.

From the users' point of view, a virtual representation of the laboratory is needed. In this case study, two RELATED views will provide them with the visual (virtual and real) representation that will let users conduct the remote setup interactively.

RELATED development steps

Real-time services

These were developed using the RT Series System of LABVIEW.[24] This software allows the design of robust real-time applications embedded on several real-time targets. In this case, a NI PCI-7030/6040E real-time multifunction I/O board has been used, which has a 486/133 MHz processor dedicated to real-time operations. The development cycle is quite simple: A vi file with the real-time operations (DAQ operations and math operations to compute the control actions based on a PID algorithm) is implemented. Now, this code (vi file) is downloaded into the real-time target, and it is executed. After this, a LABVIEW application (another vi file) is developed to communicate with the real-time target and to provide TCP/IP communication streams (which will be used in a RELATED module). These two vi files are shown in Figs 1 and 2.

At this time, the servo-motor set-up is already available to be connected to the RELATED framework. A RELATED software module adapter can now be 'produced' which will have access to the data defined in the real time services. Such adaptation consists of endowing the code with an interface (based on TCP streams) to access the values of the variables, using a protocol to exchange information. In this way, real-time control details are hidden to the lab developer.[19] At the end of the paper, Fig. 9 illustrates the XML definition of the module.

RELATED views

Two views have been developed to provide users' basic interactivity with the laboratory. The first (Fig. 3(a)) provides a virtual representation of the motor set-up, i.e., displaying the plant state while the experiment is being conducted. The second view (Fig. 3(b)) shows the temporal evolution of SPEED, POSITION, and CONTROL variables in many signal scopes. Also, both views let users tune PID parameters

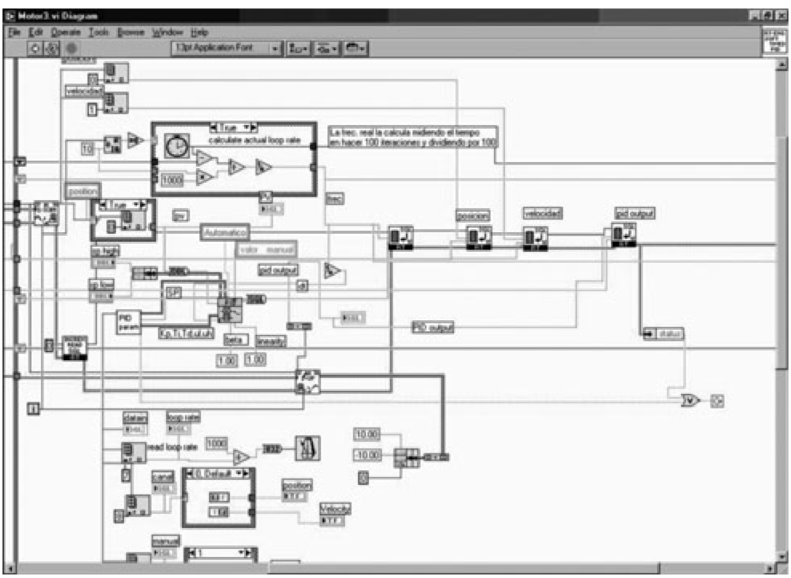


Fig. 1 *Vi file embedded into the real time target.*

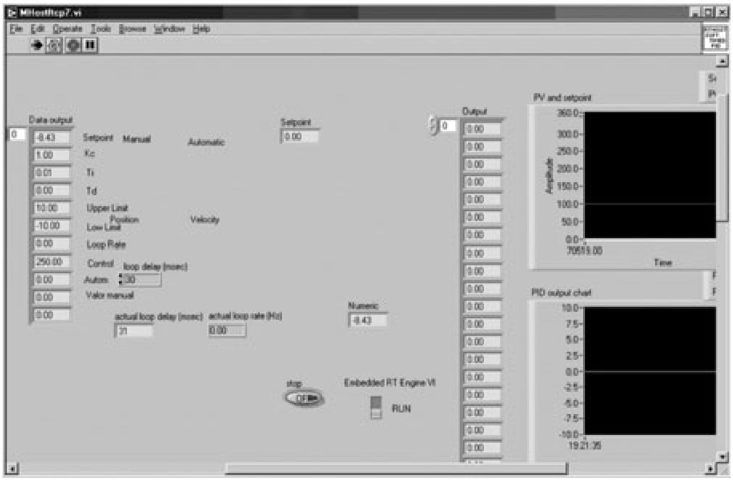
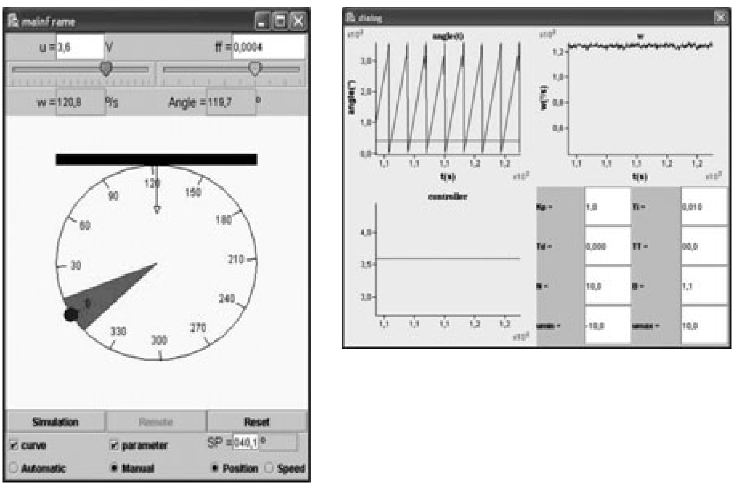


Fig. 2 *Vi file providing TCP/IP communication streams.*



(a) (b)

Fig. 3 (a) *Virtual representation of the motor*; (b) *Trend Graphics and PID parameter textboxes*.

(Fig. 3(b)), switch to manual/automatic control mode, and choose control of position or speed (Fig. 3(a)).

The second RELATED view (Fig. 4) defines the video channel representation. It is used to display a live image of the servo motor, letting users adjust the video parameters to receive a fine stream of video (the user receives visual feedback in terms of frames per second). Also, PTZ (Pan, Tilt and Zoom) camera functions are added as follows:

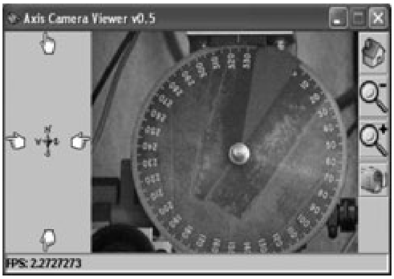
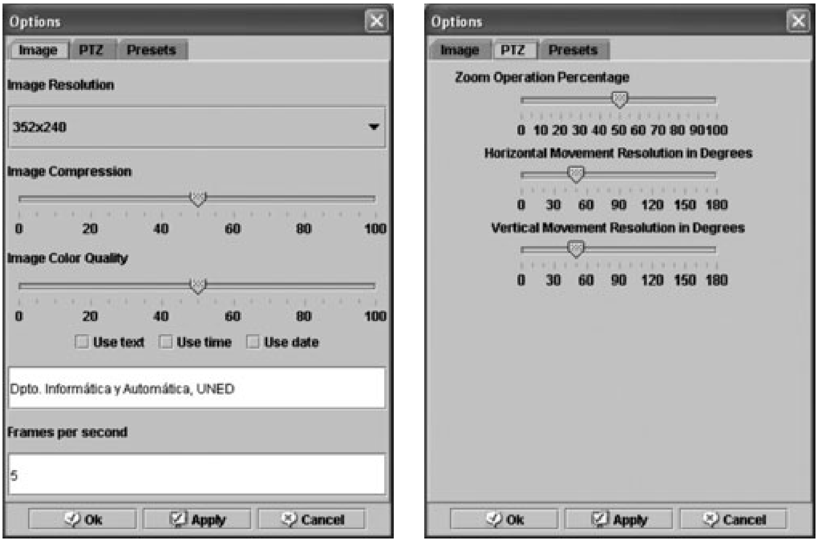


Fig. 4 *Video channel representation.*

(a)



(b)

Fig. 5 (a) *Image options*; (b) *PTZ options*.

- To move the camera horizontally and vertically.
- To zoom the image in and out.
- To move the camera to the initial position.
- To start and stop the recording of experiments.

Using the options menu of the image panel, the user has the possibility of adjusting the video stream quality using the options panel (Fig. 5(a)) and thus to change the image resolution, the JPEG compression, etc. The PTZ tab (Fig. 5(b)) allows the user to change the percentage/degrees 'ratio' for use in the Zoom and Movements operations. Also, a new movement function is included to centre the image by clicking with the mouse on any position of the video image.

This view needs the execution of a module (video server module) that operates as an image server and allows distribution of images to the video RELATED view. In Fig. 9 you have the XML definition of the views and the video server module.

Experiments

Several experiments can be done using this remote/virtual setup. For example, Fig. 9 shows the definition of two very common experiments in this kind of didactical setup: control of speed, and control of position. When the experiments are running, the state of the motor is shown in Figs 6 and 7. In both cases, to change the PID values interactively or to switch to simulation mode, the simulation button (Fig. 3(a)) can be used.

With the remote motor setup it is possible to view the real angular position by means of the zoom-in operations and, thus, to have a better and larger image. As

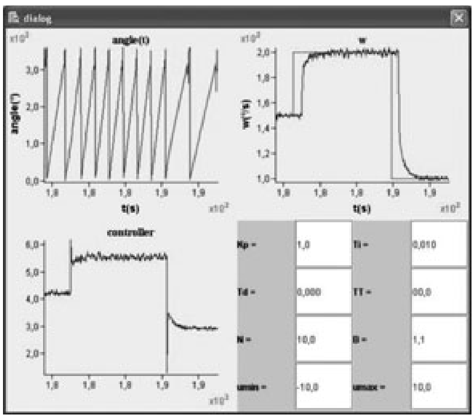


Fig. 6 *Virtual view of the speed control experiment.*

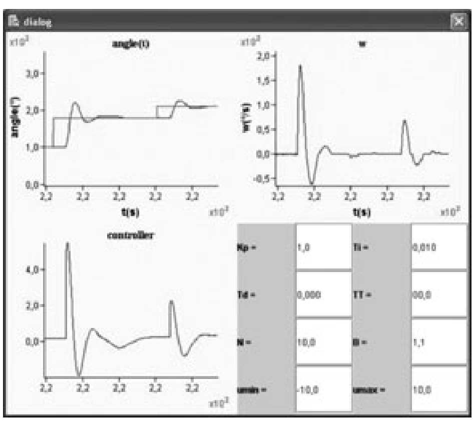
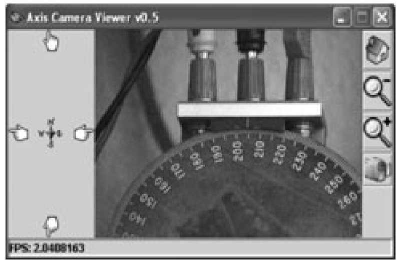


Fig. 7 *Virtual view of the position control experiment.*

(a)



(b)

Fig. 8 (a) *Virtual angle*; (b) *real angle*.

can be appreciated from Fig. 8, the reference angle (210 degrees) has been reached, but the real angle (in the remote servo) is near to 205 degrees. Calibration of the sensors will be needed to adjust the virtual representation (data from sensors) and the real-angle visualisation. Maintenance operations can be carried out remotely in order to update the laboratory day-by-day.

Conclusions

Nowadays, the development of virtual and remote labs can be considered a mature technology. There are many well-known approaches and solutions to providing students with remote access to simulations or didactical setups across the Internet. However, the construction of every virtual and remote lab is afforded by each university group on its own; it is not a collaborative and shared activity. In general, each group does not reuse the software or hardware developed by others.

The RELATED framework is being designed as a new tool, oriented to facilitate the construction of educational lab networks for 'hands-on' experience in control engineering. Three words can summarise its main features: reusability, simplicity, and flexibility. This new XML-based approach allows educators to create new Internet-based control labs using legacy code and control elements regardless of location. In this way, RELATED helps control engineering departments to solve two important problems: the absence of room for didactical setups, and the lack of financial resources.

```xml
<?xml version="1.0" encoding="utf-8" ?>
- <system name="SERVOMOTOR" type="0">
  Servo Motor Remote System Definition
  - <module name="SERVO module">
      Remote data access module
      <param name="ExecutionTime" type="long" value="200">Thread time</param>
      <var name="SPEED" type="double" vector="yes" initial="0" max="245" min="0" units="V">Speed of motor</var>
      <var name="POSITION" type="double" vector="yes" initial="0" max="360" min="0" units="V">Position of motor</var>
      <var name="CONTROL" type="double" vector="yes" initial="0" max="10" min="0" units="V">Speed of motor</var>
      <var name="SETPOINT_VECTOR" type="double" vector="yes" initial="0" max="10" min="-10" units="V">Vector for position or speed setpoint for servo motor</var>
      <var name="SETPOINT" type="double" initial="0" max="10" min="-10" units="V">Position or speed setpoint for servo motor</var>
      <var name="KP" type="double" initial="1.0" max="2" min="0" units="%">Proportional gain</var>
      <var name="TI" type="double" initial="0.0" max="2" min="0" units="seconds">Integral time</var>
      <var name="TD" type="double" initial="0.0" max="1" min="0" units="N/A">Derivative time</var>
      <var name="SPEED_OR_POSITION" type="boolean" initial="false" max="1" min="0" units="N/A">Type of control</var>
      <var name="MANUAL_AUTOMATIC" type="boolean" initial="true" max="1" min="0" units="N/A">Open or close loop for control</var>
      <implementation type="JAVA" jarfile="file://d:/RELATED/ServoModule/classes/servomodule.jar" classname="servo">Access to servo</implementation>
    </module>
  - <module name="VIDEO SERVER module">
      Image video server used in conjuntion with Image Panel View
      <param name="ExecutionTime" type="long" value="100">Thread time</param>
      <var name="IMAGE" type="rawbytes" vector="yes" initial="0" max="0" min="0" units="V">Motor Image</var>
      <implementation type="JAVA" jarfile="file://d:/RELATED/VideoServerModule/classes/videomodule.jar" classname="VideoServer">Video Server</implementation>
    </module>
  - <view name="Image Panel" jarfile="file://d:/RELATED/VideoPanel/video.jar" classname="AxisViewer">
      Axis Camera Panel view
      <use name="IMAGE" module="VIDEO SERVER module" as="image" />
    </view>
  - <view name="Virtual" jarfile="file://f:/RELATED/EJS/ejsview.jar" classname="EJSView">
      Virtual motor
      <use name="SPEED" module="SERVO module" as="w" />
      <use name="POSITION" module="SERVO module" as="angle" />
      <use name="CONTROL" module="SERVO module" as="controller" />
      <use name="SETPOINT_VECTOR" module="SERVO module" as="reference_w_angle" />
      <use name="SETPOINT" module="SERVO module" as="reference_scalar" />
      <use name="KP" module="SERVO module" as="Kp" />
      <use name="TI" module="SERVO module" as="Ti" />
      <use name="TD" module="SERVO module" as="Td" />
      <use name="SPEED_OR_POSITION" module="SERVO module" as="Speed_Control" />
      <use name="MANUAL_AUTOMATIC" tmodule="SERVO module" as="Manual_or_Automatic" />
    </view>

  - <experiment name="Speed control" sampleTime="100">
      Change PID values for speed control
      <duration type="Time" time="180" />
      3 minutes experiment
    - <run module="SERVO module">
        This module is necessary for data accesss
        <set name="SPEED_OR_POSITION" time="0" value="true" />
        <set name="MANUAL_AUTOMATIC" time="0" value="false" />
        <interactives names="KP,TI,TD,SETPOINT,SPEED_OR_POSITION,MANUAL_AUTOMATIC" show="true,true,true,true,true,true" />
      </run>
      <run module="VIDEO SERVER module">This module is necessary for image viewing</run>
      <open view="Image Panel" />
      <open view="Virtual" />
    </experiment>
  - <experiment name="Position control" sampleTime="100">
      Change PID values for position control
      <duration type="User">Infinite time (depends on global resource planning)</duration>
    - <run module="SERVO module">
        This module is necessary for data accesss
        <set name="SPEED_OR_POSITION" time="0" value="false" />
        <set name="MANUAL_AUTOMATIC" time="0" value="false" />
        <interactives names="KP,TI,TD,SETPOINT,SPEED_OR_POSITION,MANUAL_AUTOMATIC" show="false,false,false,false,false,false" />
      </run>
      <run module="VIDEO SERVER module">This module is necessary for image viewing</run>
      <open view="Image Panel" />
      <open view="Virtual" />
    </experiment>
```


Fig. 9 *XML definition file according to the RELATED specification.*

In this paper, a case study has shown how the behaviour of a real remote laboratory can be represented using the basic definitions that RELATED provides. Also, it has been demonstrated that it is possible to reuse common code in order to get a fast and easy integration of different control systems. With this aim, the behaviour of a servo motor has been modelled with XML tags, and the RELATED tools have been used for the remote execution of experiments. Also, two common experiments (control of speed and position) for a didactical setup like the servo-motor have been presented.

Acknowledgements

This work has been supported by the Spanish CICYT under grant DPI2001-1012.

References

1 P. Horacek, 'Laboratory experiments for control theory courses: A survey', *Annual Reviews in Control*, **24** (2000), 151–162.
2 D. Poulis and A. Pouliezos, 'Computer assisted learning for automatic control' in *Proc. IFAC Symposium on Advances in Control Education*, Istanbul, Turkey, 1997, pp. 181–184.
3 D. Cooper and D. Fina, 'Training simulators enhance process control education' in *Proc. ACC*, San Diego, USA, 1999, pp. 997–1001.
4 C. Röhrig and A. Jochheim, 'Java-based framework for remote access to laboratory experiments', in *IFAC/IEEE Symposium on Advances in Control Education*, Gold Coast, Australia, 2000.
5 S. E. Poindexter and B. S. Heck, 'Using the web in your courses: What can you do? What should you do?', *IEEE Control Systems Magazine*, **19**(1) (1999), 83–92.
6 A. Bhandari and M. H. Shor, 'Access to an instrumental control laboratory experiment through the World Wide Web', in *Proc. ACC*, Philadelphia, USA, 1988, pp. 1319–1325.
7 C. Bohus, L. A. Crowl, B. Aktan and M. H. Shor, 'Running control engineering experiments over the Internet', in *Proc. 13th IFAC World Congress*, San Francisco, USA, 1996, pp. 25–33.
8 P. A. Arpaia, A. Baccigalupi, F. Cennamo and P. Daponte, 'A remote measurement laboratory for educational experiments', *Measurement*, **21**(4) (1997), 157–169.
9 J. Sánchez, F. Morilla, S. Dormido, J. Aranda and P. Ruipérez, 'Virtual control lab using Java and Matlab: A qualitative approach', *IEEE Control Systems* Magazine, **22**(2) (2002), 8–20.
10 J. W. Overstreet and A. Tzes, 'An Internet-based real-time control engineering laboratory', *IEEE Control Systems Magazine*, **19**(5) (1999), 19–34.
11 W. Sheng, L. Choo-Min and L. Khiang-Wee, 'An integrated Internet-based control laboratory', in *IFAC/IEEE Symposium on Advances in Control Education*, Gold Coast, Australia, 2000.
12 C. Ko, B. Chen, V. Ramakrishnan, Y. Zhuang, S. Hu and J. Chen, 'An innovative advance in conducting experiments in virtual laboratories in the National University of Singapore', in *IFAC/IEEE Symposium on Advances in Control Education*, Gold Coast, Australia, 2000.
13 K. Ling, Y. Lai and K. Chew, 'An online Internet laboratory for control experiments', in *IFAC/IEEE Symposium on Advances in Control Education*, Gold Coast, Australia, 2000.
14 J. Sánchez, S. Dormido, R. Pastor and F. Morilla, 'A Java/Matlab-based environment for remote control system laboratories: illustrated with an inverted pendulum', *IEEE Trans. Educ.*, **47**(3) (2004), 321–329.
15 C. Schmid, 'Remote experimentation techniques for teaching control engineering', in *4th International Scientific-Technical Conference (Process Control 2000)*, Kouty nad Desnou, Czech Republic, 2000.
16 D. Gillet, H. A. Latchman, Ch. Salzmann and O. D. Crisalle, 'Hands-on laboratory experiments in flexible and distance learning', *J. Engineering Education*, April (2001), 187–191.

17 D. Gillet, Ch. Salzmann and P. Huguenin, ‘A distributed architecture for teleoperation over the Internet with application to the remote control of an inverted pendulum’, in *Lecture Notes in Control and Information Sciences 258: Nonlinear Control in the Year 2000*, Vol. 1 (Springer, London, 2001), pp. 399–407.
18 C. C. Ko, B. M. Chen, J. Chen, Y. Zhuang and K. C. Tan, ‘Development of a Web-based laboratory for control experiments on a coupled tank apparatus’, *IEEE Trans. Educ.*, **44**(1) (2001), 76–86.
19 Ch. Salzmann, D. Gillet and P. Huguenin, ‘Introduction to real-time control using LabVIEW™ with an application to distance learning’, *Int. J. Engng Ed.*, **16**(3) (2000), 255–272.
20 R. Pastor, J. Sánchez and S. Dormido, ‘RELATED: A framework to publish Web-based laboratory control systems’, in *IFAC Workshop on Internet-Based Control Education*, Madrid, Spain, 2001.
21 R. Pastor, J. Sánchez and S. Dormido, ‘A XML-based framework for the development of web-based laboratories focused on control systems education’, *Int. J. Engng Ed.*, **19**(3) (2003), 445–454.
22 A. V. Nguyen, Y. Rekik and D. Gillet, ‘Integrated environment for Web-based experimentation in engineering education’, in *ED-MEDIA 2002*, Denver, Colorado, 2002.
23 R. Pastor, J. Sánchez, S. Dormido, C. Salzman and D. Gillet, ‘Integrating a Web-based laboratory into a reusability-oriented framework’, in *IFAC/IEEE Symposium on Advances in Control Education*, Oulu, Finland, 2003.
24 National Instruments, LabVIEW Real-Time Module. Information available at *http://sine.ni.com*.