

# Evolving towards better architectures for remote laboratories: a practical case

Javier García-Zubía, Diego López-de-Ipiña, Pablo Orduña\*

\* Faculty of Engineering, University of Deusto, Avd Universidades 24, 48007 Bilbao (Spain)

**Abstract**— A WebLab is a remote laboratory controlled via Internet. Traditionally, the focus on WebLab design has been placed on the hardware side and the communication link between the controlling PC (WebLab server) and the hardware prototype. Little attention has been paid to the other communication segment going from the WebLab server to the remote users' PCs, since this has been regarded as a "solved software problem". Consequently, aspects such as security, scalability, accessibility, or user friendliness have often been disregarded in WebLabs. This situation may be solved if a serious effort is placed on the definition of proper distributed software architectures for WebLabs. In this paper, we describe such ideal software architecture, resulted from an iterative process seeking a web-based, secure, scalable, multi-user, multi-device WebLab.

**Index Terms**—Remote laboratories

## I. INTRODUCTION

Present Western Society has nothing to do with the past in several fundamental aspects as family life, integration of handicapped persons, labour organization, new technologies in entertainment and work, distance learning, time schedules, etc. All of these aspects make it necessary for our habits to change. The University must live this change and try to offer the services that Society is demanding at every instant.

It has always been an objective of universities to decentralize part of their activities and to encourage group work or collaborative work: bring the university everywhere and anytime. Moreover, the European Union is taking up these aspects under the Bologna Declaration. In this new educational framework, students will have more freedom to organize their time, education will be less strict as far as timetables are concerned and consequently, the organization of the laboratories will have to adapt to these changes even to the extent of being more complicated.

The concept of WebLab has been around since the early nineties. Its development is widespread in laboratories of analog [1] and digital [2] electronics, programmable logic [3] or process control [4]. We can encounter good examples of WebLabs in different countries: USA [5], Colombia [6], Spain [3,7], Portugal [8], Italy [9], Corea [4] and so forth.

A WebLab can be studied from different points of view:

- *Didactical*: didactic goals, quality and suitability of the WebLab, didactic platform integration, etc. [9,10,11,12]
- *Hardware technology*: cards, electronic prototype, data acquisition, etc. [6,10]
- *Software technology*: client/server design, security, integration, etc. [10]
- *Software development platforms*: Web-Services [4], LabView [2,13], C applications [14], JAVA [15], Matlab [8], etc.
- *Communication*: through RS-232[16], TCP/IP[11], XML[17], etc.
- *Social*: international solidarity, disabled people adaptation, etc. [3].

The recent popularity of the WebLab concept, its different approaches and the abundant existing bibliography only prove the great activity on a field which is called to represent a cornerstone of worldwide engineering education.

WebLabs are traditionally designed by electronic and control engineers who naturally tend to place a major attention on the hardware side of the system. They usually follow a three step process: (1) choose a programmable device, (2) attach it to a server, accessible through the web or simply a TCP/IP socket, and (3) design a simple protocol to record programs in the remote device, send inputs and receive outputs. Unfortunately, the software side involved in the last two steps is often paid too little attention and hence a poor usage of the remotely available programmable hardware devices is achieved. We believe that better software architectures for WebLabs should lead us to more user-friendly, cost-efficient, reliable and scalable WebLabs.

The structure of the paper is as follows. In sections II and III we review the WebLab concept and its advantages. Section IV shows the different existing WebLab technologies. Section V shows the evolutions of the software architecture of our WebLab and proposes a new "ideal" architectural model for WebLabs which will allow among other things collaborative work. Sections VI and VII describe the current incarnation of our WebLab at the University of Deusto and its results. Finally, section VIII draws some conclusions.

## II. WHAT IS A WEBLAB

At present, student practices are carried out in laboratories, with all the problems inherent to them, organization and specialized equipment. There are two solutions or complements frequently used: simulators and

virtual laboratories that “reproduce” the real ones but are actually software, that is, the student reproduces his own laboratory in a computer.

There is a third solution: the so called remote WebLabs, from now on WebLabs. In this case, students access via TCP/IP the hardware equipment and programs, and this way they monitor and control the real evolution of his practical case through a Webcam or by any other means. Now we can say that the student appears to be in the laboratory, although he may be at home or anywhere.

### III. ADVANTAGES OF REMOTE WEBLABS

The design and use of a remote WebLab in a Faculty of Engineering has the following clear advantages:

- Better performance for the lab equipment since they are available to students during 24 hours a day and 365 days a year.
- Organization of laboratories. It is not necessary to keep the labs open at all times, only have operative the WebLabs.
- WebLabs promote collaborative work.
- Organization of work for the student. With WebLabs both students and lecturers can better organize their own time, including class time schedules.
- Autonomous learning. WebLabs promote autonomous work; fundamental in the new European Higher Education Space.
- Open to Society. WebLabs open the laboratories to Society.
- Distance courses. WebLabs permit the organization of engineering courses without the need to have the students present, avoiding many of the current problems.
- Integration of handicapped students. Since all the hardware equipment is controlled by a computer, they may be used by handicapped students with software / hardware techniques specially designed for their particular needs.

### IV. DIFFERENT WEBLAB TECHNOLOGIES

A WebLab is composed of different and clearly defined parts:

- Some laboratory equipment to control. This equipment has to be programmable in order to permit its control by means of software, for example PLC, numeric control, microcontroller, DSPs, FPGAs, etc.
- A server that will have the laboratory equipment connected. This computer will be in charge of programming the devices and controlling it.
- A website (i.e. a set of web pages on a web server) offering Internet access to the people interested in using the lab remotely.
- A set of client computers connecting to the remote lab server through the WebLab site.
- A Webcam attached to the WebLab server, providing images through the WebLab site in order to show the evolution of the practical case.

- A custom made card that will establish the communication between the server and the prototypes or laboratory equipment.

The main design strategies for WebLabs are explained in the next sections. In all of them the sequence of operation turns out to be the same:

- The device to program must be physically connected to a PC which will act as a server. This PC must contain and run the server program whose mission is to track possible connections to clients through the Internet.
- The students connect to the WebLab site by means of their usernames and passwords.
- Once the connection and authentication process has been completed, the student may begin his session with the WebLab. His first step will be to upload the program that has been designed to control the prototype.
- After the student program is recorded on the target prototype, he will proceed to activate the inputs and observe the behaviour of the prototype by means of the Webcam. At this point the student will be able to see if his practical case has been completed correctly, that is, the prototype has evolved correctly or as it should have. If everything turns out correctly the student may leave the service of the WebLab, or in case it has not turned out as it should, he will reprogram the WebLab until he successfully completes the exercise.

#### A. WebLab based on a specific Client/Server TCP/IP application

In this case, the design of the remote client first implies the elaboration of a Client/Server program using the TCP/IP protocol [3]. In general the computer language used is C. Programming this application could be complicated since the control of all the operations falls within the scope of the activities of the programmer, on the other hand, this programmer has the total control of the application and must do everything. The application has two parts: the client and the server parts.

Fig. 1 shows the case of a programmable logic device, but if the device to be reprogrammed were a PLC, things would not change very much, the only difference would be that the student would have to reprogram the .bat file in order to run the PLC program.

The problems with this strategy would be:

- Quality and maintenance of the application strongly depend on the programmer.
- All the security strategies fall on the programmer and this is a powerful and at the same time risky business.
- Any change in the hardware, in the TCP/IP protocol, the use of primitives, etc., will cause a change in the code.
- All of the above conditions advise us that the specific Client/Server strategy should only be used in specific and particular cases.

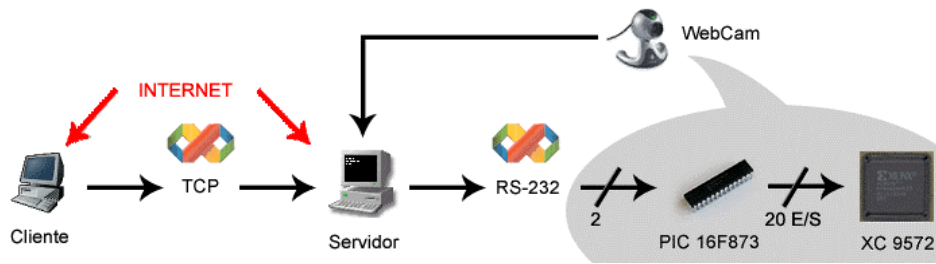


Figure 1. WebLab for programmable logic based on a Client/Server application

**B. WebLab implemented as a Web application**

The objective is still the same, but the strategy varies. Instead of designing a Client/Server application, where the programmer is responsible for everything, here the programmer must use as many of the standard services offered by Internet as possible, for example:

- Management of security will be the responsibility of the operating system or the server instead of the programmer.
- Access to the service will be via Web, that is, the student will access a Web page and will not have to execute the Client program at his console.
- Communication will be under the control of the “internet services” (HTTP) and the same will be true for error recovery.
- Management of the login will be the responsibility of the server.
- Interoperability of operating systems will fall upon themselves.

Consequently, in this case the designer has the responsibility of building all the above mentioned services for the Web page, worrying about the users and their profiles instead of the services associated with them. The quality of these services is the responsibility of the operating system or Internet. For example, the security

policy will be the same as the one used in the whole enterprise and will not have to be particularized.

Fig. 2 shows the general scheme of the WebLab application. The solution shown uses microservers (implemented in microcontrollers, microprocessors, FPGA, etc.) as a bridge between the server and the programmable device. This new solution presents the following advantages:

- WebLab changes from a hardware service to a Web service, in fact, the laboratory may be seen as an intranet.
- The microserver has an IP, consequently the whole communication with the hardware equipment is done via Internet in an IP local network or even an intranet.
- Since the devices have an IP, this allows us to grow in number without having to modify the physical network. In other words, we can control as many devices as we may need, the only thing we would need is new jacks (Internet access points) and their corresponding IP address.
- It is possible to create Web services over the programmable devices without having to modify the server.
- Control devices could intercommunicate, improving the quality of the WebLab.

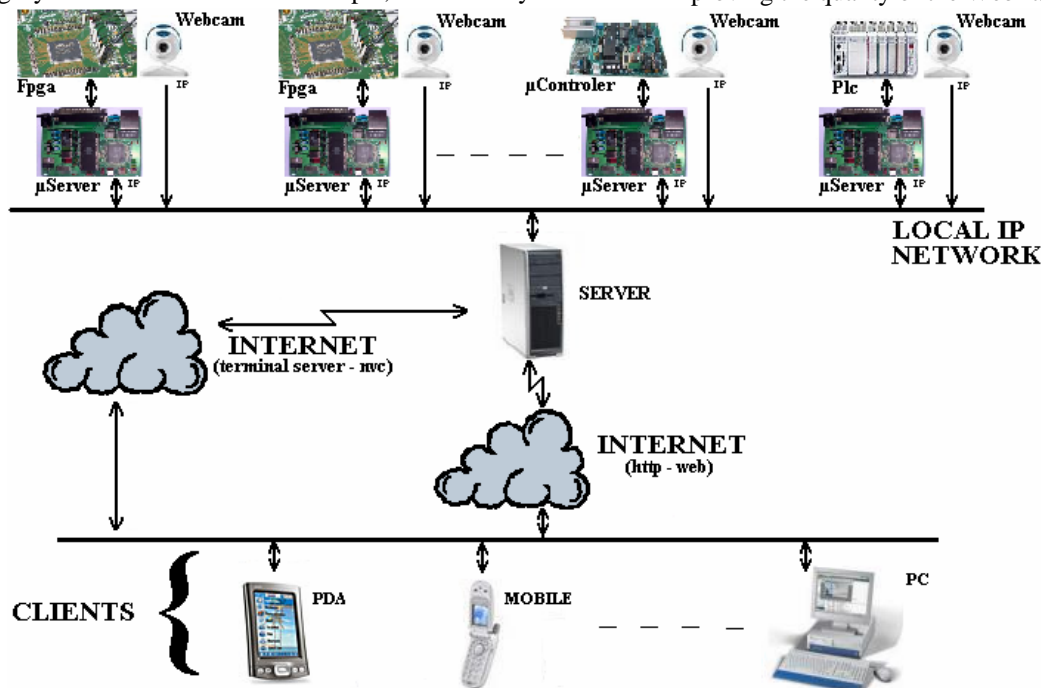


Figure 2. Structure of a WebLab as a microserver-based Web application

C. WebLab implemented as a Windows Terminal Server or similar

This strategy is based on the use of the Terminal Server facility of the Windows operating system (or another similar application such as VNC) [7]. The basic idea behind these tools is to enable a user to control the applications running in a remote PC as if he was sitting in front of that PC. The terminal server application only displays the remote PC's graphical interface on the user's PC and propagates the keyboard and mouse events from the user's PC to the remote PC. The sequence turns out to be the following:

- The student connects himself to a Web page, introducing his username and his password.
- Once accepted, Windows turns control over to the student's server. In other words, the keyboard and the screen belong to the student, but the controlled machine is the server. The previous Web page is only used to capture the key events but does not organize the work of the client as in the case of a WebLab based on a Web application.
- It is now that the student begins to control the server and can upload new software to the controlled device.
- Once the student finishes his practice, he leaves the Terminal Server.

Under this strategy, the designer does not have to deal with the Client/Server communication, since it is done by means of the Terminal Service facility (or by means of the VNC application). The risk is evident: the student "could delete files" or undertake any hazardous action on the remote machine. However, the Terminal Server allows for the definition of user profiles. Depending on who makes the connection, the access rights to the files and/or programs of the server will be different. These profiles also contain information on whether a user may or may not delete files, reconfigure the server, etc.

The difference between this strategy and the previous one is that in the latter we can organize and control the actions of each client in the server in a personalized way, that is, it can better control the way the user operates. On the other hand, with the Terminal Server the student is free to work on his own within his profile. Moreover, the student must have this service installed in his own computer.

D. WebLab implemented under other strategies

As it has already been mentioned, there are several ways of implementing a WebLab, specially now that there is such a wide diversity of software, for example:

- Implementations based in real time distributed environments, for example, CORBA. The main advantage in this case is that we have at our fingertips the power of the CORBA environment, but perhaps this could be also a disadvantage.
- Implementations based in LabView [10]. This is a widely used solution, and the main advantages are their power, knowledge and availability in university environments already oriented to the design of WebLabs. Its main drawback is that

there is little free software available and the cost of it is really quite high.

V. SOFTWARE ARCHITECTURE EVOLUTION IN WEBLAB-PLD

In the Faculty of Engineering of the University of Deusto we have been developing the WebLab-PLD for four years. The system implemented (see Fig. 3 and 4) is a remote laboratory to control programmable devices of the PLD type, specifically CPLDs of Xilinx. The WebLab-PLD allows the student to: (1) upload a new jedec file, (2) change the inputs and (3) see the outputs from a Webcam. If the results are correct the practice finishes, otherwise the student has to repeat the process.

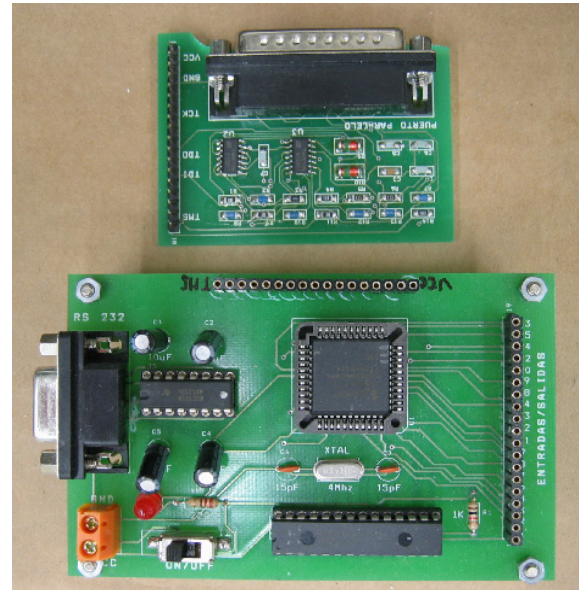


Figure 3. University of Deusto's WebLab-PLD



Figure 4. Aspects of the WebLab-PLD

In this section we illustrate the successive improvements applied to the software architecture of our WebLab to progress from a 1-1-1 (1 user, 1 server, 1 programmable device) to an N-1-N WebLab. Our final goal is to achieve a reliable, cost-efficient, user-friendly, collaborative and scalable WebLab. Progressively we examine the advantages and disadvantages of the different iterations of our software architecture to conclude with a definition of what we consider may be the canonical architecture for WebLabs.

Whereas section IV was a general description on WebLab strategies, in this one we shift the focus to

explain the evolutions suffered by our architecture and the problems encountered in that process.

The software architecture of our WebLab has gone through the following four iterations:

1. Socket and Applet-based Proprietary solution [3].
2. Web-based solution [11].
3. AJAX-based Web solution [11].
4. MicroServer-based AJAX-based Web solution.

As a result of this iterative process we have envisioned the architecture of a next-generation WebLab which will allow mainstream access to WebLabs worldwide, we have called this architectural concept “Universal WebLabs”.

#### A. Socket and Applet-based Proprietary Solution

Fig. 5 shows the first iteration of the software architecture we devised for our WebLab. A proprietary standalone client implemented in C communicated using the SDLnet library with the WebLab server. This server was in charge of communication through RS-232 with a PIC acting as bridge of a programmable PLD. In parallel to the command-line application remotely controlling the programmable device, an ActiveWebcam applet provided by PySoft was used to observe in real-time the status of the hardware being programmed. The WebLab server kept user-access and usage control. Each time only one user could be accessing the remote device. This was a prototype only used by lecturers and guests.

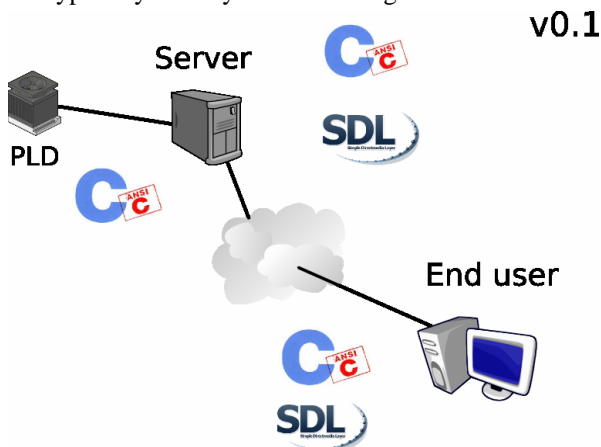


Figure 5. 1<sup>st</sup> Iteration Software Architecture

The main drawbacks of this solution were:

- *Interoperability issues.* Both the client and server solutions could only be run on the MicroSoft Windows platform.
- *User-friendliness issues.* The users needed to start two independent applications, the controlling standalone C-based application and the Java viewing applet. Moreover, the controlling client offered a primitive command-interface through which FTP-like commands could be used to upload new logic to the programmable device, induce inputs and read outputs.
- *Security issues.* On the server side, the firewall had to be configured to enable traffic offer two non well-known ports rather than using already opened ports such as 80 for HTTP. In addition, there was not built-in user access control. Consequently, there was fear to open the WebLab

to the public, and it was only used for demonstration purposes within the University’s LAN.

#### B. Web-based Solution

Fig. 6 shows the second iteration of our software architecture. Here, the server-side was composed of three elements: a) an Apache web server hosting a webpage with the controlling and viewing applets, b) a Python server which communicates through the serial port with a PIC that controls a PLD and c) a Webcam server broadcasting the images captured. In this iteration, the client application was totally based in Java, accessible through a web browser with a pre-installed Java plug-in. The controlling applet communicated with the controlling server, whereas the viewing applet connected with the Webcam server.

The WebLab server’s logic was updated to keep user-access and usage control. Each time only one user could be accessing the remote device for a maximum period of time (120 secs). The only requirement imposed to students was to use a browser with a pre-installed Java plug-in.

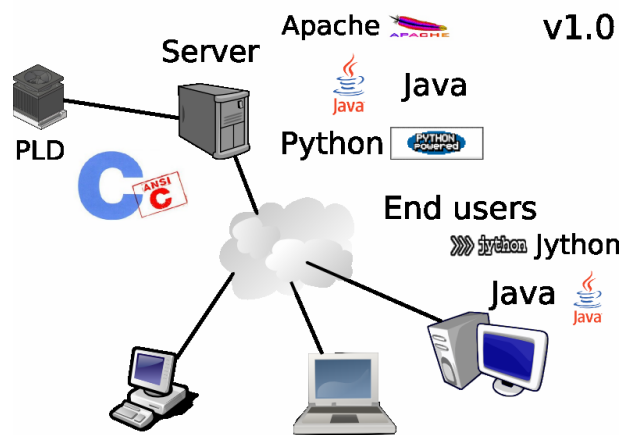


Figure 6. 2<sup>nd</sup> Iteration Software Architecture

This solution still presented some issues regarding user-friendliness and security:

- *User-friendliness issues.* We had two independent applets executing on the same webpage. The download of the applets took some time and required the user browsers to have installed the Java plug-in.
- *Security issues.* A security alert was raised every time the user downloaded the controlling applet since this required access to the file system of the user in order to upload a file with the new programming logic. Moreover, we still had to keep opened two ports in the firewall: one for the webcam server and another for the controlling server. This supposed a hassle for the firewall maintenance.

With this iteration, students of the “Programmable Logic” subject were given access to the system from an Internet browser outside the University.

#### C. AJAX-based Web Solution

The third iteration of our WebLab, currently in use, is shown in Fig. 7. A single client application shown in the user’s browser communicates with the server through

HTTP. We now have a web-based firewall-safe system programmed with AJAX (Asynchronous JavaScript and XML). The main benefit of this web development approach is that it only uses tools readily available on any web browser, i.e. XHTML, DOM and JavaScript. Therefore, no plug-in installations were required on the users' browsers.

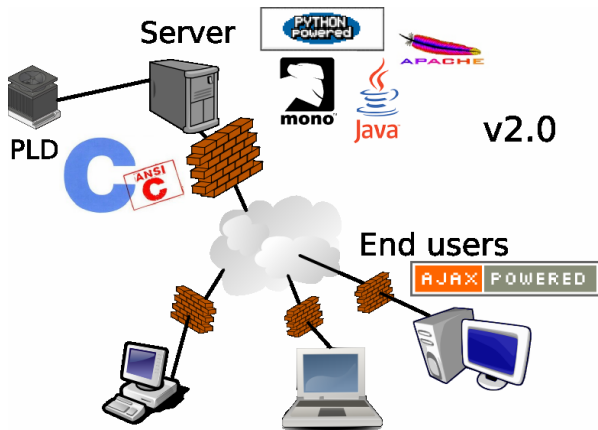


Figure 7. 3<sup>rd</sup> Iteration Software Architecture

In this iteration, the server side is composed of the following elements: a) a Java server continuously capturing images from a Webcam and saving them into a directory exported by an Apache web server, b) a Python server controlling the communication with the programmable device and c) an ASP.NET application based on Mono and running on the Apache Web Server offering a web-service interface to client applications.

The client application is now a pure HTML/JavaScript solution which follows the AJAX web interaction model, i.e. rather than changing the full content of a page every time there is an interaction between the client and server, only the portion of the page affected is modified. This technology is being applied successfully on sophisticated web applications such as Gmail, Google Maps or Flickr. A remarkable benefit of using this technology is that the control commands, responses and images are transmitted asynchronously, without interrupting the user interaction with the system, by means of the JavaScript's XMLHttpRequest object [18].

The data exchanged between the AJAX client and the Mono-based server is through the standard Web Services transport protocol, namely SOAP. The Mono-based server delegates the arriving web-service method invocations to the Python server controlling the programmable device. The latest captured image is continuously being retrieved through HTTP by the AJAX-based client by accessing to a well-known URL.

The main drawbacks of this solution are:

- *Interoperability issues.* Although the client-side is multi-platform, the server software still relies on the Windows platform. Both the serial communication and program recording programs only run on Windows.
- *Server Software Maintenance issues.* Far too many technologies are used on the server side: Java, Python and ASP.NET. For maintenance purposes it would be interesting to concentrate all the functionality in a single component developed with only one programming technology.

- *Scalability issues.* The server only provides service to one user accessing the remotely programmable device each time. Ideally we would like to network N devices controllable by the same server instance, and accessible simultaneously by N users.
- *Image Streaming issues.* The reception of the remotely programmable device images is still far from optimum. Each image is transmitted as a JPEG file instead of as a video stream which would allow for a more up to date and reliable tracking of the remote device's activities.
- *Security issues.* This iteration still lacks a semantic verification of the programs uploaded to the programmable device which would prevent the upload of hazardous software. However, now only port 80 is used in the communication between the client and server side of the system. Therefore, this solution is firewall-safe.

#### D. MicroServer AJAX Web-based Solution

We are currently progressing to the WebLab architecture shown in Fig. 8. This solution will be web-based, firewall-safe, more scalable (will provide several programmable devices) and support cooperative work among group members. N groups of users from any client platform will be able to access simultaneously to any of the N networked programmable devices.

In our third WebLab iteration, the communication and control of I/O was performed through RS-232 (see Fig. 9) by means of a PIC microcontroller acting as a bridge between the server and the electronic prototype. Moreover, the Webcam was connected to the server by means of an USB port. Therefore, if we wanted a single server to control several prototypes and Webcams we would need several serial and USB ports together with the corresponding coordination protocol for all those devices.

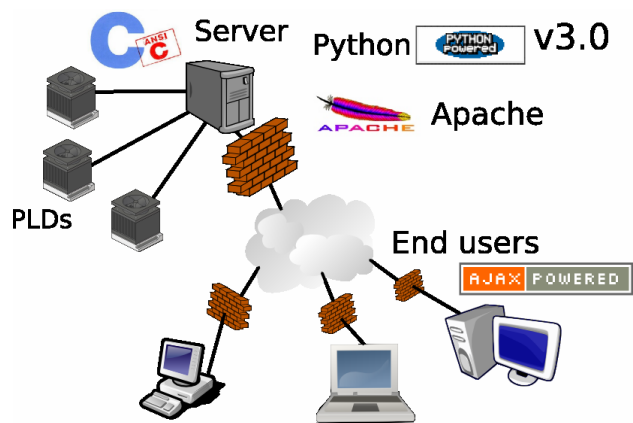


Figure 8. 4<sup>th</sup> Iteration Software Architecture

In our third WebLab iteration, the communication and control of I/O was performed through RS-232 (see Fig. 9) by means of a PIC microcontroller acting as a bridge between the server and the electronic prototype. Moreover, the Webcam was connected to the server by means of an USB port. Therefore, if we wanted a single server to control several prototypes and Webcams we would need several serial and USB ports together with the corresponding coordination protocol for all those devices.

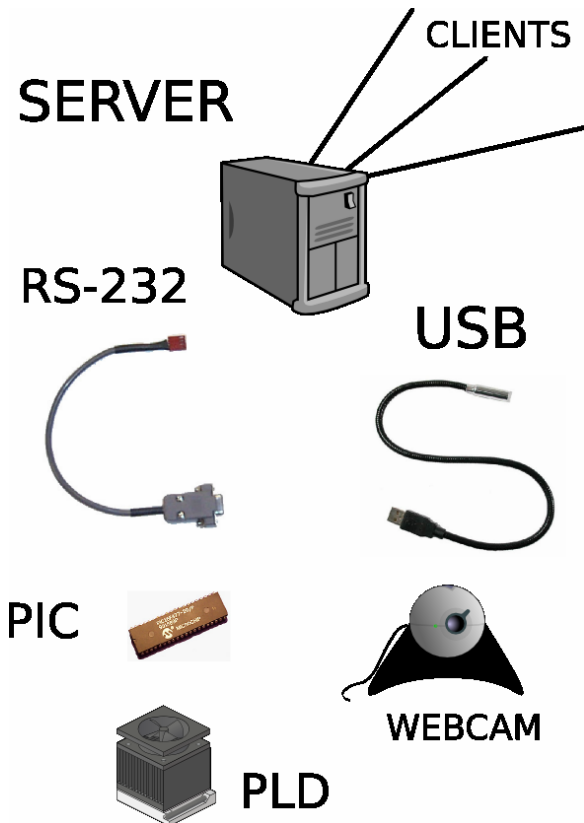


Figure 9. 3rd Iteration Electronic Prototype Connectors.

In the currently ongoing development of the fourth iteration of our WebLab we will replace the PIC microcontroller by an assortment of IP-accessible MicroServers and the USB-connected webcams by IP webcams, as shown in Fig. 10. The adoption of microcontrollers and IP webcams will turn our WebLab into a much more flexible and scalable distributed system:

- The WebLab server will no longer have to deal with the low-level RS-232 communication details. It will instead communicate through HTTP by means of data encoding standards as XML.
- The MicroServers will allow the set of programmable devices within a WebLab to be connected in a LAN. The MicroServers will connect either through an Ethernet port or will host an IEEE 802.11 chip to allow them to be wirelessly connected among themselves and the controlling WebLab server.
- The electronic prototypes attached to the MicroServers will also be capable of exchanging information among themselves. The information does not only flow between the prototype and the server, but it also can flow among prototypes with the help of the MicroServers.

With the incorporation of MicroServers each programmable device in a WebLab will be transformed into a networked node. Therefore, network administrators will now have to deal with a new type of device and ensure it is operational on a 24x7 basis.

An interesting application of this more scalable WebLab (now we can have N students simultaneously accessing to the N available programmable devices) is that its use could be shared with organisations external to our

University. For instance, taking into consideration the hour zone differences between Spain and South America, our WebLab could be accessible to South American Universities during Spanish night hours. That activity would not suppose a big disadvantage for our students, since their use of the WebLab is very marginal at night.

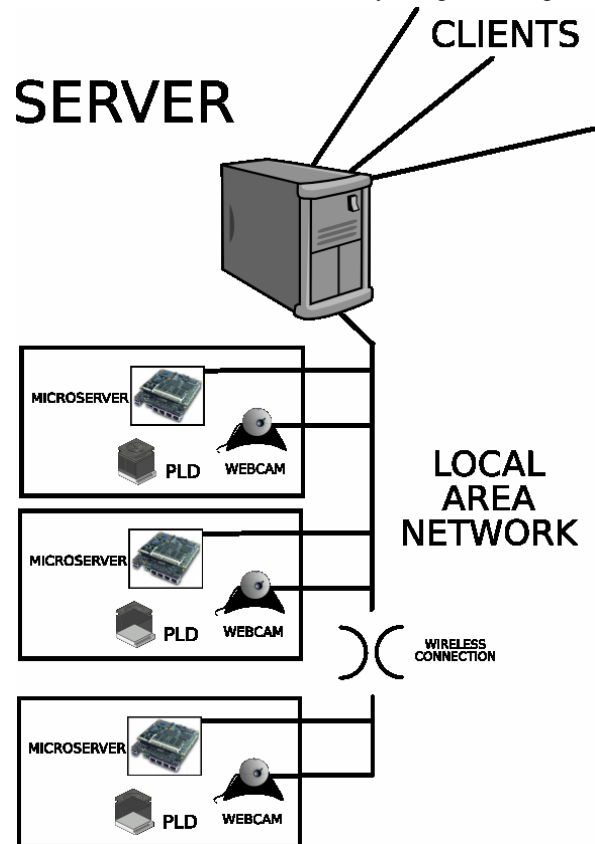


Figure 10. 4th Iteration Electronic Prototype Connectors.

On the client-side, the current AJAX-based solution will be improved to add groupware, i.e. capability of working in group, and better image retrieval features. On the other hand, the WebLab server will concentrate all the functionality currently dispersed in three components: the webcam server, the controlling Python server and the Web Services hosted in Apache. It will be implemented in a single programming language (probably Python). This server will now communicate with N autonomous MicroServers, small hardware devices attached to each programmable device, providing two main function: (1) writing programs and configurations into the remote devices, induce inputs and capture outputs, and (2) be accessible through TCP/IP. Each MicroServer will implement a cut-down web server. Access control to each of these MicroServers will be regulated by the WebLab server. In addition, we will incorporate IP cameras also accessible through TCP/IP, without having to attach them to a PC.

In conclusion, this fourth iteration will provide us with a cross-platform, secure, collaborative multi-user multi-device solution, which maximises the use of the hardware resources allocated. The core idea behind our fourth iteration architecture will be to “push away” from the WebLab server all the functionality specific to a given programmable device. In our opinion, this solution will approach to the ideal software architecture for a WebLab.


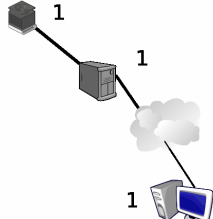
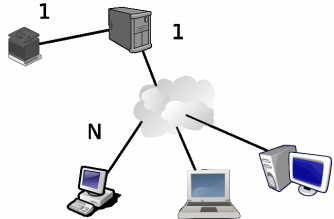
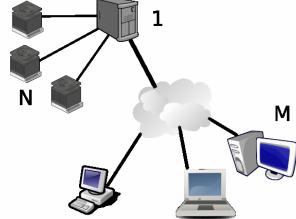
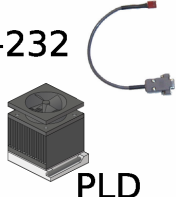

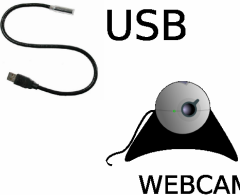
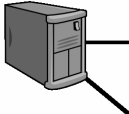
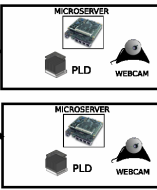








 WebLab	0.1	1.0	2.0	3.0
Device Server Client Proportion				
Connection with devices	RS-232 	SERVER 	USB 	SERVER  Network 
Client side technology				
Server side technology				
Protocol	proprietary		SOAP	
Does it use HTTP for transporting everything?	No		Yes	
Data protection	-			

Figure 11. Evolution of WebLab-PLD.

*E. Towards Universal WebLabs*

In this section we explain our vision on the future of WebLab architectures.

The adoption of IP-accessible MicroServers giving access to attached programmable devices removes the previous location dependency between the server and its associated programmable devices. Before, the server and the electronic prototypes had to be co-located in the same lab premises. However, now they can be placed in any



network accessible location. For instance, university A may implement programmable logic prototypes and host the WebLab server whilst university B may provide an assortment of process control prototypes, still controlled by university A's server.

There is a clear analogy between the World Wide Web and MicroServer-based Multi-Device WebLabs. Everybody can create a new web page, store it in a web server and automatically make it accessible on the web. Likewise, any organisation could create a set of network accessible prototypes and register them with a controlling WebLab server. In essence, we could be talking about "WWW hardware".

However, before this vision can become reality is necessary to standardize the controlling interfaces offered by a WebLab MicroServer. Every compliant WebLab MicroServer should implement the same Web Service interface, so that a given WebLab server can act as proxy between the users' browser and the MicroServers controlling any kind of programmable device. More ambitiously, we could even consider the concept of "WebLab MicroServer Plug&Play". The software stored in a MicroServer could implement automatic discovery, interaction and registration mechanisms similar to the ones provided by UPnP [19]. Thus, a manual registration of each MicroServer added with the WebLab server would not be required any longer. In essence, we would be moving from a centralised (all programmable devices in one physical location) to a distributed cross-organisational WebLab [20].

Finally, from a didactical point of view, collaborative work is very desirable. The clearest example may be document sharing. Applied to WebLabs, collaborative work will enable to do the same with hardware devices. WebLabs clearly improve the possibilities to share devices in a remote way.

We have not found any references in the literature mentioning the possibilities for cooperative work opened by WebLabs. We believe that the software architecture proposed by our fourth generation WebLab presents very promising collaborative features and will truly approach to the final goal of a WebLab, i.e. to allow almost the same kind of interaction as the one achieved by a group of people working in the same physical lab.

## VI. EXAMPLES

In the Faculty of Engineering of the University of Deusto we have developed three remote WebLabs which are currently in use: WebLab-PLD, WebLab-PLC and WebLab-Pneumatic.

The WebLab-PLD described in Section IV is used to give coverage to two subjects that have about 200 students enrolled every year.

The Automatic Control Laboratory of the Faculty of Engineering of the University of Deusto has implemented two remote control systems under Terminal Server: WebLab-PLC and WebLab-Pneumatic. In WebLab-Pneumatic (see Fig. 12) it is possible to reconfigure the pneumatic control strategy of a prototype for manufacturing. The student is capable of modifying the control strategy by modifying the relation between the electro valves, the pistons and the position detectors in the manufacturing process. This prototype was constructed using FESTO devices, the pneumatic control software is

FluidSim and the communication server-prototype is based on fibre optics with the card from Easy Port.

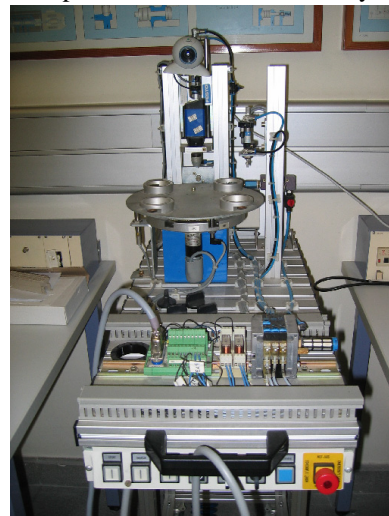


Figure 12. WebLab-Pneumatic

On the other hand, in WebLab-PLC the student may modify the start-stop strategies of the three-phase engine controlled by a PLC. In order to reconfigure the control, he only has to write a new program using Terminal Server and load it onto the PLC using the corresponding software/hardware. The final design is implemented in a Siemens PLC and the programming and recording environment is the one belonging to STEP 7 of Siemens. These WebLabs are also used in two subjects with about 300 students enrolled.

## VII. RESULTS

During the second semester of this school year WebLab-PLD was in use by the students registered in the subject "Programmable Logic". This course has 90 students registered, out of those 65 chose to do their practices using a regular laboratory and 25 have done theirs using WebLab. The results obtained are the following:

- Out of the 65 "regular" students, 16 have failed and 49 have passed, that is 75% passed and a 25% failed the practical part of the subject. Out of the 25 students using WebLab, 2 have failed and 23 have passed, that is a 92% pass and an 8% fail rate. These results clearly show that WebLabs have a beneficial effect on students, at least in our experience. In fact, during the course and in the final examination, those students not included in the WebLab experiment, complained because they said that their colleagues belonging to the WebLab experiment had more possibilities to pass the subject than they did.
- As far as the staff is concerned, we can say that WebLab is an advantage for the person in charge of the laboratories since he does not have to plan / work with those 25 students belonging to the experiment. On the other hand, WebLab has the problem of controlling the quality of the work on part of the students. For those students using the regular laboratory, the person in charge only had to look "in situ" to the student's work to assess the correctness of it. For the students participating in the WebLab experiment, the person in charge had to arrange

appointments to review their work or otherwise “accept” what the students mentioned.

- As far as hardware availability is concerned, the 25 students participating indicated that the existing WebLab was enough for all of them except in very few counts.
- The WebLab only crashed twice in the semester due to a hardware problem, namely physical destruction of the CPLD. The reason was an erroneous assignment of input/output pins on part of the student. We are at present working to find a way to filter those problems, i.e. to semantically check the students’ code.
- All the students have the same time to operate with inputs/outputs of the WebLab: 90 seconds. This time can be easily modified.
- There have been no forbidden or irregular accesses to the WebLab.

Table 1 summarises the results of a questionnaire completed by the students who used the WebLab. The grading system goes from 1 to 5.

TABLE I. RESULTS OF THE QUESTIONNAIRE

Questions	Average
1 Has WebLab helped you with the subject?	4,6
2 Did you feel that you were in a better position by being in the WebLab group?	4,7
3 Do you think it is a good idea to extend the WebLab experiment to all the students?	4,7
4 Is it easy to use?	4,4
5 How is the quality of the webcam?	3,2
6 Were you comfortable managing the inputs?	3,7
7 What do you think about the time assigned to each connection?	3,7
8 What do you think about the inputs/outputs implemented?	3,8
9 Being far from the prototype, have you felt in control of it?	4,1
10 Would you like to use WebLab in other subjects?	4,3
11 What is your global satisfaction with WebLab?	4,7

The results of the Table 1 indicate that:

- All the questions related to the WebLab in general (1,2,3, 10, and 11) are very well rated, around 4.7, i.e., students like the WebLab.
- In the questions specific about WebLab-PLD and its use (4, 5, 6, 7, and 8) the students are a little more critic, specially about the quality of the webcam used. In any case the average value is close to 4.
- Question number 9 has a special interest. Students indicate that even if they are far away from the prototype they feel in control of it, i.e., as they were at the prototype’s location using it on their own.

## VIII. CONCLUSIONS

We have presented here a way to open hardware laboratories to the outside world by means of Remote WebLabs. The most remarkable thing of this work is that there are several techniques to do this and some of them are quite easily applicable in any laboratory or educational institution that may be interested. The advantages are

several and all of them seem interesting to universities, students and Society in general.

Traditionally little attention has been placed to the software part of WebLabs. This paper has shown the benefits of aiming better software solutions. A good software solution should lead to a more efficient use of the hardware resources. Consequently, we have applied an iterative process to the software architecture of our own WebLab in order to progress from a 1/1/1 (user/server/programmable device) to an N/1/N WebLab.

As a result of our iterative study we suggest a new canonical software architecture based on the concepts of Web Services and MicroServers which presents the following features: cross-platform, secure and firewall-safe and scalable (multi-user and multi-device).

Lastly, the current incarnation of our WebLab-PLD has been positively used and evaluated by students, as proven by the results of the questionnaire undertaken.

## REFERENCES

- [1] Gustavsson, I. et al. “A flexible remote electronics laboratory”. *2<sup>nd</sup> International Symposium in Remote Engineering and Virtual Instrumentation, REV 2005*, Brasov (Romania), July 2005
- [2] Barrón, M. “Laboratorios virtuales para enseñanza por internet” en *I Jornadas de Tendencias sobre eLearning, TEL 2005*, Madrid (Spain), febrero de 2005.
- [3] García Zubía, J. “Programmable Logic and WebLab” V European Workshop on Microelectronics Education, *Proceedings of the 5th European Workshop on Microelectronics Education*, Lausanne (Switzerland) ISBN: 1-4020-2072-4, 2004, pp: 277-282.
- [4] Ko, C.C.; Chen, Ben. M.; Chen, Jianping. *Creating Web-Based Laboratories*, Springer-Verlag 2004, London, ISBN: 1-85233-837-7
- [5] Alamo, J.A., MIT Microelectronics Weblab, Marzo, 27, 2001. <http://web.mit.edu>
- [6] Pérez M. et al. “Laboratorios de acceso remoto. Un nuevo concepto en los procesos de Enseñanza-Aprendizaje”. <http://digital.ni.com/worldwide/latam.nsf/web/all/F54369A0EC8C0B4486256B5F006565A9>
- [7] Kahoraho Bukubiye, E., Larrauri Villamor, J.I. “A WebLab System for the Study of the Control and Protection of Electric Motors”, *Proceedings of Telecommunication, Electronics and Control*, pp. 7. Cuba 2002. ISBN: 84-8138-506-2, 2002.
- [8] Almeida, P., Viera Coito, F., Brito Palma, L. “An Environment for Remote Control”. *1<sup>st</sup> International Workshop on e-learning and Virtual and Remote Laboratories, VIRTUAL-LAB’2004*, Setubal (Portugal), August 2004.
- [9] Casini, M.; Prattichizzo, D. y Vicino, A. “e-Learning by Remote Laboratories: a new tool for control education” . *The 6th IFAC Conference on Advances in Control Education*, Finland, 2003.
- [10] Cabello, R. et al. “EMERGE: A European Educational Network for Dissemination of OnLine Laboratory Experiments”. *Innovations 2004*, Ed. INNER, 2004.
- [11] García-Zubia, J et al. “A new approach for implementing remote laboratories: a practical case”. *2<sup>nd</sup> International Symposium in Remote Engineering and Virtual Instrumentation, REV 2005*, Brasov (Romania), July 2005
- [12] Soysal, O. “Computer Integrated Experimentation in Electrical Engineering Education over Distance” *Proceedings of ASEE 2000 Annual Conference*, Saint Louis, MO, June 2000.
- [13] Gomes, C. “Distance Learning Remote Laboratories using LabVIEW”. *1<sup>st</sup> International Workshop on e-learning and Virtual and Remote Laboratories, VIRTUAL-LAB’2004*, Setubal (Portugal), August 2004
- [14] Aliane, N.; Martínez, A.; Fraile, A.; Ortiz, J. “LABNET: laboratorio remoto para control de procesos”. *Actas de las XI Jornadas de Enseñanza Universitaria de la Informática, JENUI*

- 2005, Madrid (Spain), ISBN: 84-9732-421-8, Julio 2005, pp: 515-522,.
- [15] Pelcz, A. et al. "Remote experiments using JAVA: Implementations in the Virtual Electro Lab project". *1<sup>st</sup> International Workshop on e-learning and Virtual and Remote Laboratories*, VIRTUAL-LAB'2004, Setubal (Portugal), August 2004
- [16] Gomes, L.; Costa, A. "Embedded systems introductory course supported by remote experiments ". *1<sup>st</sup> International Workshop on e-learning and Virtual and Remote Laboratories*, VIRTUAL-LAB'2004, Setubal (Portugal), August 2004.
- [17] Bagnasco, A.; Chirico, M.; Scapolla, A.M. "XML Technology to Design Didactical Distributed Measurement Laboratory (RmwLAB) Instrument", *IEEE Transactions on Instrumentation and Measurement*, VOL. 54, N° 1, February 2005
- [18] McLellan, D. "Very Dynamic Web Interfaces", O'Reilly Xml.com, <http://www.xml.com/pub/a/2005/02/09/xml-http-request.html>, February 2005.
- [19] The Universal Plug and Play Forum, 2005, <http://www.upnp.org/>.
- [20] Rasche, A. et al. "Distributed Control Lab". *1<sup>st</sup> International Workshop on e-learning and Virtual and Remote Laboratories*, VIRTUAL-LAB'2004, Setubal (Portugal), August 2004.

#### AUTHORS

**J. García-Zubia** is with the Faculty of Engineering of the University of Deusto, Bilbao, Spain (e-mail: [zubia@eside.deusto.es](mailto:zubia@eside.deusto.es)).

**D. López-de-Ipiña** is with the Faculty of Engineering of the University of Deusto, Bilbao, Spain (e-mail: [dipina@eside.deusto.es](mailto:dipina@eside.deusto.es)).

**P. Orduña** is student at the Faculty of Engineering of the University of Deusto, Bilbao, Spain (e-mail: [pablo@ordunya.com](mailto:pablo@ordunya.com)).