Hands-On Remote Labs: Collaborative Web Laboratories as a Case Study for IT Engineering Classes

Mario A. Bochicchio, *Member*, *IEEE*, and Antonella Longo, *Member*, *IEEE*

Abstract—The development of a reusable collaborative software framework for the remote control of a large range of laboratory equipments is an interesting topic to teach information technologies in an engineering school. The design and implementation of this kind of framework, in fact, requires the ability to integrate skills about software engineering, computer networks, human-computer interaction, distributed architectures, and remote control of hardware devices (i.e., laboratory equipments). In the paper, we describe our experience in the development of a reusable framework for remote laboratories, which has been adopted as a case study in two different scenarios at our university.

Index Terms—Collaborative learning tools, computer science education, virtual labs, Web-based interaction.

1 Introduction

Laboratory classes play a crucial role in engineering schools. Good pedagogical reasons, such as illustrating and validating analytical concepts, introducing students to professional practice and to the uncertainties involved in nonideal situations, developing skills with instrumentation, and developing social and teamwork skills in a technical environment motivate the need for their inclusion in the curricula [1].

On the other hand, laboratory management can be resource-intensive and expensive, since it requires qualified staff and continuous equipment maintenance and evolution, so that the number of laboratories is often limited, also due to economic factors. Also, for these reasons, the adoption of alternative access modes (e.g., remote laboratories) is more and more considered by universities [2].

Remote labs, in fact, can extend the capability of a conventional laboratory by increasing the number of times and places a student can perform experiments [4], [5] and extending its availability to several students [6]. Moreover, they have the potential to provide affordable experimental data by sharing expensive laboratory equipments within a larger pool of users [3].

Nevertheless, remote laboratories are not a straightforward solution:

1. They require space, devices, and maintenance, like or more than hands-on labs.
2. They are usually perceived as "less effective" because the experience is mediated by digital computers and networks and by distance, i.e., experimenters obtain data by controlling geographically detached equipment.

From the effectiveness point of view, comparative studies show that students are motivated and willing to work in remote labs [2], [6] and most of them think that remote labs are more effective than schoolbooks and also than digital simulators [7]. For these reasons, since 1996 [8], remote laboratories have been increasingly popular, but their development has mostly been driven by technical aspects rather than by the needs to reproduce a satisfactory user experience.

Most of them, for example, have been designed for a single user at a time, so that students engaging in remote learning are isolated. Moreover, current solutions are very "specific" (i.e., they are tailored for a given lab equipment), and do not support high-level concepts like reuse (to be adopted for a whole range of laboratory equipments), user privileges, and security features (in order to prevent users from accomplishing some tasks that might result in potential equipment damage).

In order to overcome the described problems, we foster the research of a more general, scalable, and reusable approach, not limited to a specific laboratory equipment or to a specific experiment. Our goal is "not to reinvent the wheel" each time a device goes online, but trying to define some guidelines for a standard approach in the design of remote labs, with the aim of permitting the definition of satisfactory and effective user experience. This point is even more crucial when we need to put online different lab equipments or to interact with a number of remote equipments at a time (e.g., to compare two samples of the same metallurgic artifact with two remote electron microscopes at the same time, etc.).

In our opinion, Computer-Supported Collaborative Learning (CSCL) can provide remarkable support to the efficacy of this kind of remote labs. The aim is to have a networked laboratory summing up the best aspects of hands-on experiences and the best features of the Web. In

• *The authors are with the DIDA Lab, Innovation Engineering Departiment, University of Salento, via per Monteroni, 73100 Lecce, Italy. E-mail: {mario.bochicchio, antonella.longo}@unisalento.it.*

Manuscript received 31 Mar. 2009; revised 6 July 2009; accepted 31 July 2009; published online 6 Aug. 2009.
For information on obtaining reprints of this article, please send e-mail to: lt@computer.org, and reference IEEECS Log Number TLTSI-2009-03-0068.
Digital Object Identifier no. 10.1109/TLT.2009.30.

fact, we believe that collaborative aspects are essential to reinforce the interaction among the learners: The distance laboratory must foster active learning and comprehension construction by the student. In the social setting of the in-lab experience, the learner interacts directly with other students, the instructor, equipments, activities, and other elements. These interactions guide interpretation and construction of concepts. They are key aspects of the laboratory experience that must be included in distance education laboratory experiences as well [9]. As a consequence, we feel that a general purpose remote lab must be based on a synchronous collaborative environment.

In this paper, we will introduce the **We**b **Col**laborative **Lab**oratory (WeColLab) system, which is a purposely developed system, enabling the remote control of real laboratory equipments (like robotic arms, telescopes, electron microscopes, etc.); it lets groups of students share their lab experiences over the Web.

The framework has been successfully adopted for a telescope and for a robotic arm at our campus, demonstrating the reusability and scalability of the approach.

In the following, we will refer to MicroNet, that is, our Web-enabled electron microscope, and AstroNet, which is our Web-enabled telescope. MicroNet has been adopted as a case study in two different scenarios:

1. For student's internship, graduate, and PhD theses preparation in our information technology (IT) and computer engineering school. The development of a platform like MicroNet, in fact, requires the ability to integrate skills about software engineering, computer networks, human-computer interaction, distributed architectures, and remote control of hardware devices (i.e., laboratory equipments).
2. In a "Physical Metallurgy" class for engineers that has been selected because of three main reasons:
    - the remote approach allows to simultaneously use two electron microscopes from two distant sites in our campus for comparative purposes;
    - the microscope installation rooms are not suitable for educational purposes. They are in the building's basement for vibration and stability reasons;
    - the presence of a large number of students in the microscope room can produce vibrations, humidity, and other disturbances invalidating the performed measures.

Astronet has been used as a usability case study, together with Micronet.

The paper is organized as follows: Section 2 depicts the research foundation and our previous experiences, Section 3 is the description of the WeColLab platform, Section 4 describes the use of WeColLab as a case study for IT students, while Section 5 depicts the conclusions and further works.

2 Background

Remote laboratories target a large range of devices, from different scientific areas. This means that they are not restricted to a single educational topic, but they are being used for several devices and experiences that can be controlled using a computer [2], [11]. As the remote laboratory platforms are getting mature, we observe that they are still built without a shared interoperable approach. For example, Gravier et al. [11] surveyed 42 different remote laboratories finding that every project implements its own software architecture with no reuse. Nevertheless, all the analyzed projects are built on a common hardware/software structure, comprising:

- the device or the equipment to be remotized;
- a computer connected to the equipment, i.e., the gateway between the device and the remote user;
- the middleware, to interact with remote lab.

Even so, technologies vary a lot from one remote laboratory to another, which prevents reusability and interoperability among the applications.

Our research experience with remote laboratories started in 2003, when we were involved in the framework of the Telepresence Instant Groupware for Higher Education in Robotics (TIGER, a Government-funded research initiative named FIRB) project [16], which aimed at producing four prototypes of robotics experiments distributed among 11 Italian universities in a shared Electronic Learning Environment (ELE), with three main objectives:

1. possibility for the users to execute remotely "sensible" experiences on real processes mediated by the Web;
2. new and more rational management of time and space since the users can access this web service anytime and anywhere;
3. more rational management of costs in terms of laboratory equipment, of human resources and their organization.

We contributed to the project with the development of the remote control of a robotic arm, which was part of the large experimentation with students required by the project. TIGER, in particular, allowed us to study [17], [18] the issues related to telelaboratories for engineering education. During this project, as in [11], we analyzed 25 remote Web labs, finding that:

- The technologies and the programming languages generally used to connect the device and the local computer are often based on proprietary software, mainly Matlab with Simulink and LabView with data socket, with some software based on Visual Basic or Python [4], [5], [6], [7], [8], [9].
- Java is frequently used to link the laboratory equipments with the remote users, but it is often coupled to different technologies like PHP, HTML, CORBA, VRML, etc. Different approaches, based on ASP, ActiveX, Python, and C++, or proprietary software based on LabView and Matlab are also exploited by various authors [4], [5], [6], [7], [8], [9]. Some remote labs adopted software solutions, such as VNC, as it provides them with the remote control over the local computer connected to the corresponding equipment. Nonetheless, this solution was given up as it lacks security and is bandwidth-intensive [10].

- A remote lab project usually implements a stand-alone application. It can be very expensive to build, since it requires a large amount of time, money, and specific skills. The software application is usually dedicated and is not supposed to be reused in other similar labs.
- Hands-on laboratories can comprise several devices that create an experimental workbench when connected together. The experiences we have compared address only the remote control of one device at a time. In order to provide students with complete workbenches, remote laboratories need to connect different devices, which can be distributed at different places on the Web.

Moreover, the experiences described in 22 remote labs show that collaborative aspects are still little considered in remote laboratories. In particular, we point to the following shortages:

1. *Lack of presence awareness* among the participating students: this feature focuses on the concept of awareness and concrete perception about the physical presence (as in several instant messaging systems) and the activities of other participants, and their interactions with the remote equipment.
2. *Lack of user perspectives*: in a collaborative Web application, users should have different interaction experiences according to their role; for example, in a Web lab, to moderate the virtual classroom, the teacher and the tutor need widgets and indicators which are not available to students.
3. *Lack of group dynamics:* this concept takes into account how users interact with each other (i.e., the presence of a leader or a moderator, etc.) and it is very important for student's evaluation and for evaluation purposes.

3 The WeColLab System and the Electron Microscope

In order to define a Web-based system, enabling real laboratory equipments (like a telescope or a microscope with all its auxiliary devices) to be remotely controlled by a virtual class through a domestic Internet connection, we collected the following requirements:

- It must be Web-based: Students should not need any special software to carry out their experience. As any Web page, it should be linked to other Web pages and accessed by means of a standard Web browser. The implementation should be based on standard protocols and common components to guarantee the desired Web compatibility.
- It must be collaborative: Like in a multiconference, small groups of two to 20 participants (students and teachers) should be able to see/hear/talk to each other, supported by various collaboration tools (shared whiteboard, picture annotation, chat, etc.); moreover, they should see (all together) and remotely control (one at a time) the laboratory equipment.
- It must manage security features: A coordinating tutor/supervisor should manage the laboratory session, to authorize the control requests and to protect the equipment against any potentially damaging operation.
- It must be reusable over different lab types: A "standard" approach must be defined to cover a large range of computer-controlled lab equipments.
- The detailed requirements must be elicited by means of a comprehensive goal-driven approach, like KAOS [15], in order to bring all users, stakeholders, goals, and objectives out. The approach should especially consider the main features of a collaborative Web application (i.e., presence awareness, points of view, and group dynamics).

The scenario we support with the WeColLab platform is that of a virtual classroom which must remotely use a lab equipment (i.e., an electron microscope or a telescope) located somewhere in the world. The virtual classroom is made up of a lecturer (describing the lesson and commenting the experiment), a tutor (facilitating the use of the platform, supervising the correct and safe use of the equipment, solving technical problems, and moderating the discussion) and a group of students (from two to 20), all connected to the Internet at least via a domestic ADSL. The tutor owns tools to moderate the discussion, to tell disturbing students off the experiment, up till disabling their audio and video. Students can reserve their turn to remotely use the equipment and they can remotely control it. External people can optionally watch the lab session, but neither can they participate in the classroom discussion nor can they operate with the equipment.

3.1 The WeColLab Platform

According to the above requirements, we will describe the WeColLab platform, in the case of the Micronet project, which is the project for remote control of the electron microscope of our Research Department. As shown in Fig. 1, the main components of the WeColLab system are:

- **The Equipment Server** made up of the Laboratory Equipment (e.g., the electron microscope) and the Redirector, which allows the equipment to be remotely accessed and controlled, granting the proper access privileges to users.
- **The Collaboration Server** which manages the collaborative application logic, orchestrating the multimedia streams coming from the equipment server and from the users (students, tutor, teacher, and watchers). The Collaboration server is based on the MS IIS 5.1 Web Server and on the Adobe Flash Media Server 2. The component is split into two parts:
  - **The WeColLab Collaborative Engine,** which implements the multivideoconference, the shared whiteboard, the audio mixer, and all the main features of our WeColLab system.
  - **The Micronet application,** which is the only equipment-specific part of the Component. We can adapt the Collaboration Server to different remote Web labs (like a telescope, a spectrometer, etc.) by customizing this piece of software (about 20-200 lines of server-side actionscript code).

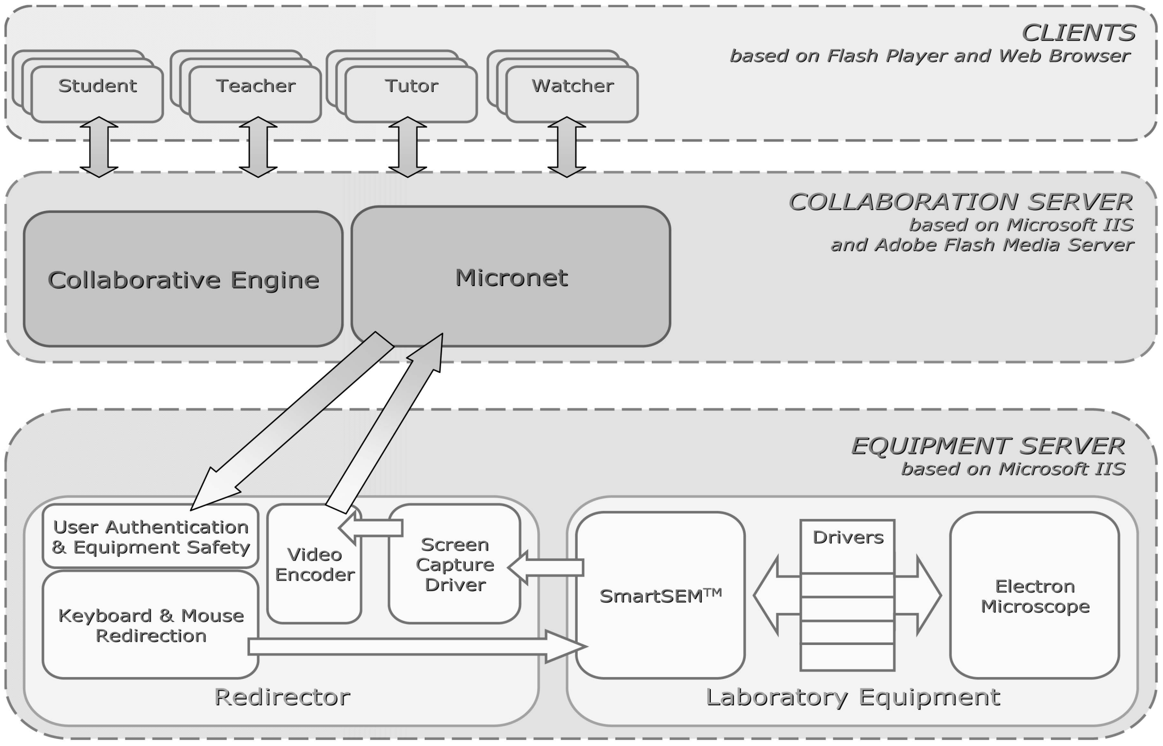


Fig. 1. WeColLab' logic architecture, specialized for MicroNet.

- **The clients** which represent the students, the teacher, the tutor, and external watchers. They receive audio, video, and images from other participants and from the Equipment Server via the Collaboration Server.

The remote control of the Equipment Server consists of three elements: a component which locally processes the commands (keystrokes and mouse coordinates) sent by the clients, another component that is able to capture the monitor video flow (with 800×600 resolution), and a third component which compresses and sends the video flows to the Collaboration Server in real time.

According to the Laboratory Equipment type, we have developed two kinds of Redirector components:

- **Software Redirector**: If the laboratory equipment includes a standard Windows-based system as controller, the Redirector components are deployed on it;
- **Hardware Redirector**: If the laboratory equipment is based on proprietary hardware, not including a standard Windows-based system as equipment controller, we add an external computer with the role of I/O redirector, which interfaces the laboratory equipment via an external keyboard and mouse emulator, for input commands, and via an XVGA video grabber for the acquisition, the compression, and the transmission of the equipment's video output. This solution is more expensive because of the additional hardware, even if technically safer.

Technically, the server-side components of WeColLab are based on the MS Windows OS, while the WeColLab clients are based on Flash components and they are played in Web browsers and are OS-independent.

The Collaboration Server is based on the streaming capabilities of Adobe Flash Media Server 2 (FMS2), which offers a unique combination of multimedia streaming features and a flexible development environment to create interactive and collaborative applications. FMS2 allows all the participants to receive and transmit their own audio/video flows, from WebCams and microphones, like in a multiconference. The video coming from the Equipment Server is also acquired, compressed, and sent in streaming to all the participants by means of FMS2.

We compared Microsoft Media Server and FMS2, but only FMS2 owns the following fundamental features for our aims:

- Compatibility with the main operating systems and browsers, which allows to reach more than 98 percent of the users connecting to the Web.
- Cheap distribution of high-quality video content, due to the efficiency of the *On2 VP6 codec.*
- Automatic client's bandwidth detection and consequent negotiation with the proper bit rate.
- Dynamic buffer setup, which reduces the startup time according to the length, the bit rate, and client connection's speed.
- Ability to capture and transmit live audio/video flows from any camera and microphone detected by the operating system and connected via a USB or Firewire port.
- Multichannel and multiuser streaming and shared object technology to synchronize data among users.
- Web administration console, allowing to show application data, objects, and audio/video streaming, and server features like CPU performance and bandwidth usage per application.

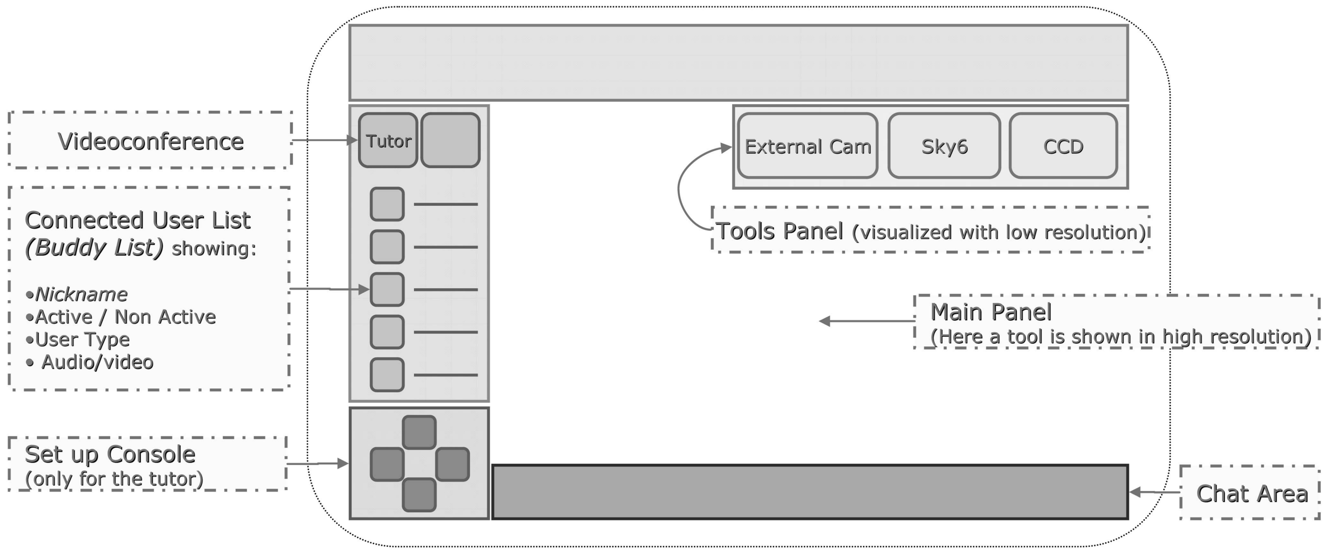


Fig. 2. Interface design of WeColLab in Astronet project.

- Robust security model to protect content, which:
  - Does not allow streaming flows to be cached;
  - Uses the RTMP proprietary transfer protocol, which reduces the stream ripping;
  - Supports the SSL protocol for data encryption;
  - Blocks connections and unauthorized access by means of its server-side programming language.

We used *Adobe Flash 8* to develop the prototypes deployed in the Flash Media Server. Instantiating the *communication components* (API for videoconference applications, connection state control, etc.) and using the proprietary object-oriented programming language (actionscript), we implemented the client applications that are able to:

1. Capture and transmit live audio/video flows from any camera and microphone recognized by the operating system.
2. Tune the bandwidth for any sent or received audio/video flow.
3. Exploit the remote shared objects technology to synchronize data among the users.

In order to capture the Equipment Server's video output, we chose a virtual camera like the TechSmith Camtasia or the *VH Screen Capture driver (*www.hmelyoff.com), which allow the optimization of the video screen capture flow, with frame rate control, the compatibility with Flash, the control of the screen region to capture, and the screen capture according to the mouse position.

In order to remotely control the equipment, we have developed three components:

1. a *server-side* module, written in Flash, hosted in the *Redirector*, which captures the *equipment server* screen video flow, compresses it, and sends it to the *Flash Media Server*;
2. a client-side application, written in Flash, which allows the remote user to see the Equipment Server screen, thanks to the Redirector, and to send the keystrokes, the mouse coordinates, and the mouse events to the Redirector, under the supervision of the tutor;
3. a Keyboard and Mouse Redirector, written in actionscript and in Visual Basic 6, which synchronously intercepts, receives, interprets, and executes the remote user commands (mouse coordinates, mouse events, and keystrokes).

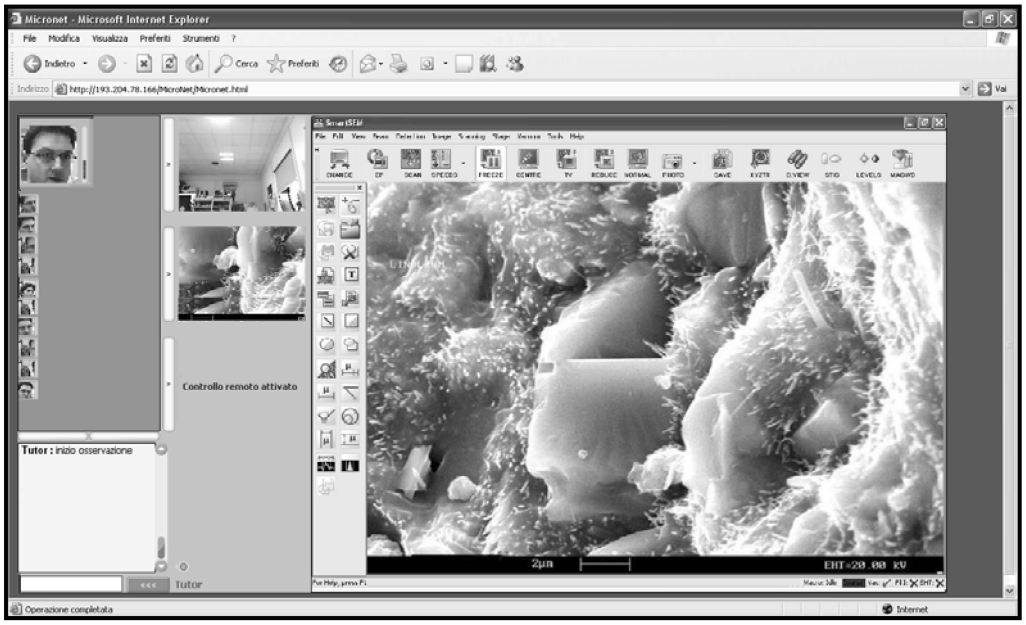
The applications have been developed in MS Windows OS because equipment servers are often Windows-compliant; Windows is also the most diffused and used OS in domestic configurations.

The choice of the FMS2 to manage all the audio/video flows requires that every client must be equipped with Adobe Flash Player. The wide diffusion of this plug-in, available for every operating system and Web browser, and installed on most of the computers connected to the Internet, makes WeColLab easily available to those who want to participate in the remote laboratory sessions.

3.2 The User Interface

Fig. 2 shows the WeColLab user interface design, customized for MicroNet. The design includes five tasks areas: on the left-hand side there is the multivideo conference area, showing the video (thumbnail size) coming from each participating students (Buddy list). The magnified videos of the tutor (in the upper part) or of the student who has been enabled to pilot the microscope are shown in the upper-left corner. The setup console (lower-left corner) is used by the tutor to tune the student's audio, to enable/disable the student's microphone, and to supervise the actual audio and video bandwidth measures.

The right-hand side of the screen contains the area with the Tools Panel in the upper part, and the Collaboration area at the bottom. The Tools Panel is organized with thumbnails each displaying different equipment views of the lab environment. In MicroNet project, we have had three thumbnails, with video streams from the lab, one coming from an ambient camera to show what happens in the lab room, another coming from a *head-mounted camera*, showing


Micronet - Microsoft Internet Explorer
File Modifica Visualizza Preferiti Strumenti ?
Indietro
Cerca
Preferiti
Indirizzo http://193.204.78.166/MicroNet/Micronet.html
Vai
SmartSEM
CHANGE EP SCAN SPEEDS FREEZE CENTRE TV REDUCE NORMAL PHOTO SAVE XYZTR D.VIEW STIG LEVELS MAGWD
Controllo remoto attivato
Tutor
2µm
EHT=20.00 kV
For Help, press F1
Operazione completata
Internet

Tutor : inizio osservazione

Fig. 3. Screenshot of “MicroNet”application.

the microscope workbench, and a third one from the Equipment Server remote desktop, reproducing the *Smart-SEM* application which is the application managing the electron microscope. The “main panel” is used to magnify one of the small-size video flows shown in the tools panel.

The collaboration area is used for a chat room.

Fig. 3 shows the screenshot of the MicroNet project where a tutor is lecturing about the features of a linoleum sample, displayed in the central part of the screen by the SmartSem application.

At the left side of the SmartSem application, the other two cams output are visualized which are magnified by clicking on them.

4 The WeColLab as Case Study for Engineering Students

To set the stage for the description of the use of WeColLab as a case study for interns and PhD students, we briefly describe the information technology and computer engineering curricula at the University of Salento.

The Information Technology Engineering course is part of the Undergraduate School of Engineering at the University of Salento. It is a three-year bachelor’s program, which aims at training an engineer who is able to perform tasks related to design and management in many fields of information technology. The main training objective of the degree is to achieve a solid basic knowledge, a sound scientific background, and widespread technical application skills to ensure immediate entry into the labor market. Particular emphasis is placed on helping students to develop the ability to design, construct, install, and optimize the use of electronic, informatics, automation, and communications tools and devices. The Information Technology Engineer is capable of developing the executive design of products, processes, and services, the development, the installation, and the testing of components, machines, and complex systems, and the maintenance and management of production divisions as well as the performance of procedures related to technical control, inspection, and assistance. He will operate within the wide sector of the industrial engineering, service companies, and public bodies. This course requires full-time attendance and involves classroom and laboratory activities. The main characterizing courses, among others, are Basic Automatic Control, Electronic Systems, Telecommunications Networks, Software Engineering, and Database Applications.

The Computer Systems Engineering course, instead, is part of the Graduate School of Engineering at the University of Salento. It is a two-year master’s program, which aims at the formation of computer engineers specialized in the design, implementation, and management of modern information systems, with particular emphasis on the digital convergence of computer, electronic, and telecommunication technologies, and on the development of Internet and Web-based services and applications. To this purpose, the computer engineering graduate will integrate a solid technological and scientific background with new interdisciplinary competences in the management and communication aspects of ICT. The application fields of interest include e-business (in a broad sense), online industries, Web based-information systems, multimedia, and computer graphics applications, Internet of Things, and more. The course requires full-time attendance and involves classroom and laboratory activities.

Core subjects of this course include computer science, automation, electronics, and telecommunications.

To get both the bachelor’s degree and the master’s degree, it is mandatory that students do an internship period at a private institution or a research lab, in order to complete their formation with learning by doing and applying concepts studied during the course. Internships usually are extended with the thesis graduation. Summing internship mandatory period and thesis preparation, students spend at least 375 hours (about 2-5 months of full-time equivalent) working on a project.

At our research lab on software engineering and telemedia, we host internships (from the engineering school, the social sciences school, and from external master's partnering with the university), students preparing engineering graduation theses (both for the bachelor's and the master's schools), PhD students in information engineering, external fellows working on research projects granted by national, international, and private funds.

In our everyday life at the lab, we experience the issues well described by Denning and Riehle in [19] about the software engineers approach to engineering and the need for systemic thinking in the development of software systems, "embracing hardware, user environment as well as software" [19]. So, for several years, we have included students in research projects, forming interdisciplinary/multidisciplinary working teams managed by experienced fellows. The last project of WeColLab framework has been an excellent case study to involve students in the development of a complex project requiring software engineering, distributed architectures, remote controlling of laboratory equipments, and human-computer interaction skills.

We structured the project with interdisciplinary workpackages sized for teams with two to three students. Some examples are the following:

1. The usability study of the WebColLab interface: this workpackage was assigned to a PhD student on communication sciences and four computer engineering interns.
2. The assessment of required bandwidth of WeColLab's distributed components and their allocation on the physical architecture: this workpackage was assigned to a student preparing the master's thesis, two IT engineering interns, and one computer engineering intern.
3. The development of WeColLab's interface with the lab equipment: this workpackage was accomplished by a computer engineering student preparing the thesis and four IT Engineer interns.
4. The design and prorotype development of a novel architcture for Remote Labs (WeColLab 2.0) based on Web Services, compliant with the MIT's ILab project [20]: This workpackage (still not completed) has been assigned to a PhD student, a computer engineering student as final thesis, and four IT engineering interns.

From the point of view of integration of skills and background, we found very interesting workpackages 1 and 2, which we will report and comment later in the paper.

The third one is strictly connected with the development of two specific instances of the platform. We will describe the experience with Micronet and Astronet.

The fourth workpackage is still running and we plan its conclusion before the beginning of Summer 2009.

Before starting the work, each student has been provided with a preparation kit, made up of documents about the WeColLab project, its architecture, and already developed prototypes, in order to create a common background about the framework.

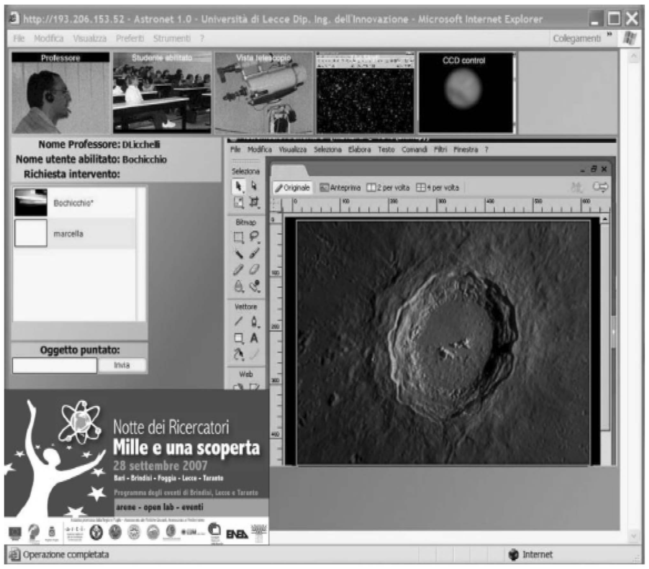

http://193.206.153.52 - Astronet 1.0 - Università di Lecce Dip. Ing. dell'Innovazione - Microsoft Internet Explorer
File Modifica Visualizza Preferiti Strumenti ?
Collegamenti
Professore
Studente abilitato
Vista telescopio
CCD control
Nome Professore: DLicchelli
Nome utente abilitato: Bochicchio
Richiesta intervento:
Bochicchio*
marcella
Oggetto puntato:
Invia
File Modifica Visualizza Seleziona Elabora Testo Comandi Filtri Finestra ?
Seleziona
Bitmap
Vettore
Web
Originale
Anteprima
2 per volta
4 per volta
Notte dei Ricercatori
Mille e una scoperta
28 settembre 2007
Bari - Brindisi - Foggia - Lecce - Taranto
Programma degli eventi di Brindisi, Lecce e Taranto
arene - open lab - eventi
Operazione completata
Internet

Fig. 4. AstroNet Webpage during a public demonstration of the project.

Moreover, each team has been involved in weekly meetings with a detailed assessment of their in-progress work and in-monthly lab meetings about the overall project and its development.

4.1 The Usability Study Workpackage

The usability study of WeColLab interface has been made during the Astronet project, which is the specialization of the WeColLab framework for remote controlling the telescope located at the Department of Physics of the University of Salento. The project, sponsored by the Regional Agency for Innovation (ARTI), involved students of secondary schools and astroamateurs in nocturnal lessons, in order to make the astronomic observation an integral part of astronomy classes.

From the user interaction point of view, AstroNet aims at simulating the experience of a group of students, who, under the guide of a teacher, choose, search, observe, and shoot pictures of different astronomical objects through a telescope, thus tracking the instrument, acquiring, and processing images.

The user interface has been designed with the intent to be usable for high school students, accustomed to videogames, chats, and videoconferences.

For Astronet, the Equipment Remote Control Area displays three views of the lab environment:

- the Sky6 application which enables the remote control of the telescope;
- the software controlling the CCD camera to shoot the astronomical pictures;
- an external camera to check the weather and the sky visibility.

Fig. 4 shows the students' interface, which differs from the tutor's one because it lacks the audio mixer controls and multiconference details.

In order to get a first idea of the usefulness and feasibility of the proposed approach, the team performed an initial user study in form of a heuristic evaluation [13].

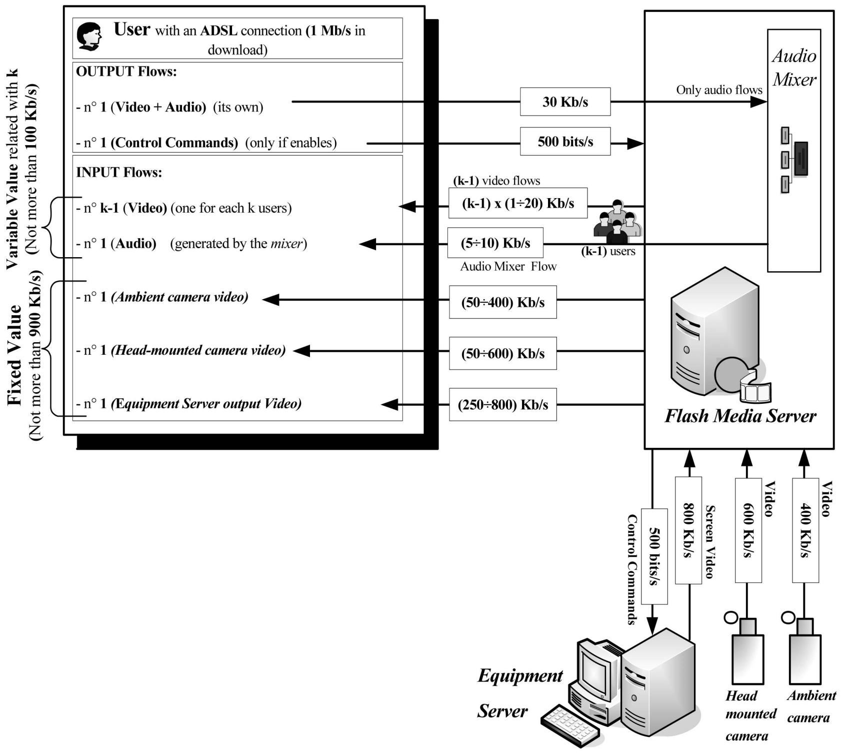

User with an ADSL connection (1 Mb/s in download)
OUTPUT Flows:
- n° 1 (Video + Audio) (its own)
- n° 1 (Control Commands) (only if enables)
INPUT Flows:
- n° k-1 (Video) (one for each k users)
- n° 1 (Audio) (generated by the mixer)
- n° 1 (Ambient camera video)
- n° 1 (Head-mounted camera video)
- n° 1 (Equipment Server output Video)
Variable Value related with k (Not more than 100 Kb/s)
Fixed Value (Not more than 900 Kb/s)
30 Kb/s
500 bits/s
(k-1) video flows
(k-1) x (1÷20) Kb/s
(5÷10) Kb/s
Audio Mixer Flow
(k-1) users
(50÷400) Kb/s
(50÷600) Kb/s
(250÷800) Kb/s
Only audio flows
Audio Mixer
Flash Media Server
500 bits/s Control Commands
800 Kb/s Screen Video
600 Kb/s Video
400 Kb/s Video
Equipment Server
Head mounted camera
Ambient camera

Fig. 5. Bandwidth usage in WeColLab.

As proposed by Nielsen [14], the PhD student, supported by the interns, evaluated the interface design based on a given set of questions related to functionality, intuition of the involved interactions, and possible problems and drawbacks. In addition, they commented on the visualization and the particular parameter settings (e.g., accessible speed ranges, audio tuning, etc.). General feedback was very positive. The usability group rated the interface design as intuitive, and easy to understand and operate. For the evaluation, they made a proper observation session of about one hour. One teammate commented that the use of multivideo conference appears very valuable for design of new types of immersive technologies. The main comments related that mixing collaboration tools, remote equipment controls, and video cameras so as to be aware of the overall scenario inside and outside the dome was an effective approach to simulate the laboratory experience. Interestingly and a little surprising for us, two users did not figure out by themselves that they are able to operate a telescope by remotely accessing the software for pointing a star, and hence, were very skeptical if the average user would be able to use the system. Actually, successfully operating a telescope requires knowledge and experience often beyond that of many high school teachers and most high school students. So, in order to evaluate the actual effectiveness, intuitiveness, and simplicity of the interface, from September 2007 to June 2008, the usability team tested AstroNet in 15 secondary classes of Apulia Region. They asked 30 students (out of 300 participants at our Astronomy night classes) to participate in the study (16 males and 14 females both at ages from 14 to 17). Most of them had experience with chats and video games and no experience in operating a telescope. Significant differences between experienced and inexperienced users as well as between male and female ones could not be observed. Each session started with a 30 minutes lesson about astronomy followed by a 30 minutes description of Astronet and sky navigation. For the evaluation, the team set up four usability tasks of increasing difficulty that had to be solved by the users operating with the system:

- **1st task:** Connect to the Web application and set up the audio system. The goal is to enter Astronet and to use the audio mixer to fine-tune the headset.
- **2nd task:** Simple interaction with the tutor and the other online students. During this task, two students and the tutor were connected through Astronet in different locations in order not to directly interfere with each other. A student listens to the other talking, chimes in, and reserves the use of the telescope
- **3rd task:** Usage of the Artificial Sky. Remotely using the Sky6 application, the student must look up a celestial object and its position in the sky, synchronizing the telescope and the dome.
- **4th task:** Shot of a celestial object picture. At first, the student shoots a picture of the celestial object pointed in the third task, using the CCD camera, evaluates the result, changes the CCD's acquisition parameters, and repeats the shot.

Since two high school students volunteered for each astronomical session, the team performed two usability tests in sequence, asking the second student not to participate in the first session and each time changing the celestial object to localize and to shoot.

At the beginning of the evaluation, users were given a short astronomical lesson and a short description of Astronet and its characteristics. Then, the Web application was handed to them to try it out themselves for five minutes. After this time, they had to solve the tasks in the given order. Following that, they were asked to give ratings about their overall impression, the interface's intuitiveness, and the offered functionality. The evaluation closed with an informal interview where users were asked some questions related to general issues, particular observations during the tests, etc. The overall duration was about 45 minutes per participant. During the study, a usability teammate sat next to the participant and took notes about what the user was doing, interesting observations, comments given during the tests, etc. Originally, the usability team wanted to implement a logging mechanism to be able to retrack all interactions. However, since this would have resulted in performance problems during the navigation, we gave up this idea. For the same reason, the time to solve the second and third task was not logged but taken with a stopwatch by the interviewer.

Since the proposed tasks were ordered with increasing difficulty, the team expected that all the users would have solved the first two but would have some troubles with the latter two, due to the implicit complexity of the astronomical topic. Surprisingly, all the students were able to reproduce all the four tasks, and have been very satisfied about the experience. All of them said the interface was very intuitive and that it was simple to use with easy to memorize functionalities. As a side result, teachers were also very satisfied because they gained high attention from students and high didactic efficacy.

Some interesting suggestions arose during the tests with students: if several students must be synchronously connected in virtual classes, a distributed audio mixer control can be very useful in order to allow the tutor to selectively control the single student's audio level; moreover, the Web application is to remote control the Sky6 and the software to shoot the celestial pictures is only able to check and direct the mouse. In order to evolve the Web system to remotely control several different software equipments, it must also get and send commands from the keyboard.

From the usability team's point of view, this experience (which involved the compuer engineering interns almost sequentially) allowed each of them to interact with people from different backgrounds: especially at the beginning, teammates' cultural background (i.e., terms, approach to problems, and sensitivity to different aspects) was not homogeneous at all and some weeks were necessary to start working together effectively.

Moreover, the interaction with the high school students and teachers forced the students team to scale down their language in order to be impressive and effective with students and teachers.

4.2 Bandwidth Assessment of WeColLab

As already said, the workpackage was performed by a team made up of a student preparing the Master's thesis, two IT engineering interns, and one computer engineering intern. They had to assess the bandwidth usage in the Micronet project.

The scenario we have designed for the use of WeColLab consists of a virtual group of k users (with $2 < \mathrm{k} < 20$), connecting contemporarily through k different domestic ADSL connections (we assume an actual peak bandwidth of at least 1Mb/s in download).

Each user sends his own live audio and video stream to the Collaboration Server and receives $(\mathrm{k}-1)$ audio live streams and $(\mathrm{k}-1)$ video live streams. This implies that the required client-side bandwidth linearly increases with the number of users. An important parameter to consider in the calculation of the bandwidth usage is the quality at which each user transmits his own audio and video, because the necessary bandwidth linearly increases (more or less) with the quality. In our case, the parameter qualifying the audio capture is the sampling frequency, measured in kilohertz. The parameters qualifying the video capture process are: the number of horizontal and vertical pixels $(\mathrm{n} \times \mathrm{m})$ per frame, the number of frames per second (*fps*), and the number of frames between two consecutive *keyframes* (*kfrs*) to use for the digital compression.

To estimate the bandwidth necessary for each user, the team made some measurements by means of purposely developed test applications adopting the following values for the camera and for the audio quality parameters: $pixels = 120 \times 90,\ fps = 6,\ audio\_sampling\_rate = 5\ kHz, kfrs = 12$. With the above values, the measured bandwidth consumption for each audio/video flow is about 30 Kb/s, where 15-20 Kb/s are for the video stream and 5-10 Kb/s are for the audio, so that the upload bandwidth for each client and for k connected users (peer-to-peer connection) is $\mathrm{k} \times (30)\ \mathrm{Kb/s}$, while the aggregate download bandwidth is $\mathrm{k} \times (30)\ \mathrm{Kb/s}$. With the same parameter values, the Collaboration Server uses a bandwidth of $(kx(k-1) \times (30))\ \mathrm{Kb/s}$.

The achieved results were clearly incompatible with the actual capabilities of domestic ADSL connection for the clients.

In order to reduce the bandwidth request per user in the multiconference system, the team proposed to adopt a client-server communication schema and developed a specific audio mixer, which receives the k audioflows and transmits in multicast only the "mixed" flow. Thus, the downloaded audioflow does not increase as a square power anymore in the Collaboration Server, but it linearly increases with the number of users.

Fig. 5 shows the estimation of the download and upload bandwidth for each user and for the Collaboration Server. It is easy to notice that the upload bandwidth is the same as before, whereas for download, one must distinguish between audio and video, because the bandwidth required for the videoflow is the same as before while the audioflow is reduced by a $(k-1)$ factor.

Now, assuming the number of connected users is equal to 20, a videoflow coming from the Equipment Server, two videoflows coming from two ambient cameras (with a total of 850 Kb/s coming from the equipment server), and a flow coming from the tutor, each user still requires 30 Kb/s in upload. Moreover, reducing the resolution and the frame rate of the videoconference so as to achieve 2 Kb/s per user, and assuming that the audioflow coming from the mixer uses 10 Kb/s bandwidth, the aggregated download bandwidth (client-side) scales down to $[30 + 30 + (18 \times 2) + 10]$ Kb/s ≈ 100 Kb/s for the multivideoconference part, and 850 Kb/s for the equipment server part. This result is compatible with the constraint of 1 Mb/s download bandwidth of domestic ADSL connections.

The team worked for four months to achieve these results. They were forced to go deep into networking issues and software architecture topics as well as analyzing offerings of networking providers for domestic connections.

5 Conclusions and Further Works

Internet-based forms of learning have been broadened to encompass live physical experiments, which, although much more complicated to implement in comparison to lecture modules, are bringing these expensive teaching tools to broader audience promoting ideas of interinstitutional joint laboratory assets.

Perfect candidates for remote laboratories are those equipments locally controlled by a computer (e.g., robotic arms and electron microscopes). The student's virtual experience, despite the limited perception confined to computer screen and the surrounding sound at lab test site [20], can simulate the in-lab one due to the virtual presence of possibilities of videoconference integrated in the remote laboratory environment. So, the drawback of a student to be in a virtual environment can be seen as a big benefit, since he/she can use the lab equipment at his own pace and time in omnipresent omnitempore fashion, but in a collaborative team that he/she decides to join.

In this paper, we have described a novel platform for remote control of lab equipment in a collaborative Web space. In particular, WeColLab is equipment-independent, thus promoting the reuse of the framework with other lab equipments, allowing the collaboration of groups of students via the Web in a multiconference session, and managing the security features.

Rather than speculate about the opportunity of using remote labs for remote experiments, we have concentrated on the use of WeColLab as a case study for IT engineering students, since its design and development requires a systemic vision of the problem and multidisciplinary/interdisciplinary skills.

As a platform, WeColLab is already evolving into a service-oriented architecture that is able to connect several remote equipments in a unique virtual workbench and several virtual classes over the Web.

Acknowledgments

The authors wish to thank interns, and master's and PhD students who worked with us in the Web **Col**laborative **Lab**oratory (WeColLab) project. We specially thank Eduardo Gira and Federica Longo for their precious contribution.

References

[1] E. Lindsay, P. Long, and P.K. Imbrie, "Remote Laboratories: Approaches for the Future," *Proc. 37th ASEE/IEEE Frontiers in Education Conf.*, 2007.

[2] J. Ma and J.V. Nickerson, "Hands-On, Simulated, and Remote Laboratories: A Comparative Literature Review," *ACM Computing Surveys*, vol. 38, 2006.

[3] D.H. Sonnenwald, M.C. Whitton, and K.L. Maglaughlin, "Evaluating a Scientific Collaboratory: Results of a Controlled Experiment," *ACM Trans. Computer-Human Interaction*, vol. 10, no. 2, pp. 150-176, 2003.

[4] G. Canfora, P. Daponte, and S. Rapuano, "Remotely Accessible Laboratory for Electronic Measurement Teaching," *Computer Standards and Interfaces*, vol. 26, no. 6, pp. 489-499, 2004.

[5] W.J. Hutzel, "A Remotely Accessed HVAC Laboratory for Distance Education," *Int'l J. Eng. Education*, vol. 18, no. 6, pp. 711-716, 2002.

[6] M. Cooper, A. Donnelly, and J.M. Ferreira, "Remote Controlled Experiments for Teaching over the Internet: A Comparison of Approaches Developed in the PEARL Project," *Proc. Ann. Conf. Australian Soc. Computers in Learning in Tertiary Education (ASCILITE '02)*, 2002.

[7] E. Scanlon, C. Colwell, M. Cooper, and T.D. Paolo, "Remote Experiments, Reversioning and Rethinking Science Learning," *Computers and Education*, vol. 43, nos. 1/2, pp. 153-163, 2004.

[8] B. Aktan, C. Bohus, L. Crowl, and M.H. Shor, "Distance Learning Applied to Control Engineering Laboratories," *IEEE Trans. Education*, vol. 39, no. 3, pp. 320-326, Aug. 1996.

[9] J.L. Leitner and W.J. Cane, "A Virtual Laboratory Environment for Online IT Education," *Proc. Sixth Conf. Information Technology Education*, pp. 20-22, Oct. 2005.

[10] J. Trevelyan, "Experience with Remote Laboratories in Engineering Education," *Proc. 14th Ann. Conf Australian Assoc. Eng. Education*, 2003.

[11] C. Gravier, J. Fayolle, B. Bayard, M. Ates, and J. Lardon, "State of the Art About Remote Laboratories Paradigms—Foundations of Ongoing Mutations DIOM Laboratory," *iJOE*, vol. 4, no. 1, pp. 18-25, Feb. 2008.

[12] A.B. Diop and J. Harms, "Remote Real Laboratory: Linux Installation and Configuration," *Proc. Int'l Conf. Internet Computing (IC '03)*, pp. 23-26, June 2003.

[13] J. Nielsen and T.K. Landauer, "A Mathematical Model of the Finding of Usability Problems," *Proc. Conf. Human Factors in Computing Systems (INTERCHI '93)*, pp. 206-213, 1993.

[14] J. Nielsen, "Jakob Nielsen's Alertbox, Why You Only Need to Test with 5 Users," http://www.useit.com/alertbox/20000319.html, Mar. 2000.

[15] A. Van Lamsweerde, "Requirements Engineering: From Craft to Discipline," *Proc. 16th ACM SIGSOFT Int'l Symp. Foundations of Software Eng. (FSE '08)*, invited paper for the ACM SIGSOFT Outstanding Research Award, Nov. 2008.

[16] D. Fabri, C. Falsetti, S. Ramazzotti, S. Longhi, P. Forcheri, L. Vismara, M.A. Bochicchio, M.G. Celentano, G. Oriolo, P. Fiorini, A. Tornambè, G. Casalino, A. Buretta, and A. Vaccai, "Deliverable 1.1. Architettura Funzionale del Sistema," Telepresence Instant Groupware for Higher Education in Robotics (TIGER), 2003.

[17] D. Fabri, C. Falsetti, S. Ramazzotti, and T. Leo, "Robot Control Designer Education on the Web," *Proc. IEEE Int'l Conf. Robotics and Automation (ICRA)*, special session on education, pp. 1364-1369, Apr. 2004.

[18] M.L. Corradini, G. Sammarco, A. Manni, and G. Parlangeli, "VRL, a Novel Environment for Control Engineering Practicing: An Application to a Fault Tolerant Control System," *Proc. Int'l Conf. Control, Automation, Robotics, and Vision (ICARCV '06)*, pp. 1-6, 2006.

[19] J.P. Denning and R.D. Riehle, "The Profession of IT—Is Software Engineering Engineering?" *Comm. ACM*, vol. 52, no. 3, pp. 24-26, 2009.

[20] V.J. Harward, J.A. del Alamo, S.R. Lerman, P.H. Bailey, J. Carpenter, K. DeLong, C. Felknor, J. Hardison, B. Harrison, I. Jabbour, P.D. Long, T. Mao, L. Naamani, J. Northridge, M. Schulz, D. Talavera, C. Varadharajan, S. Wang, K. Yehia, R. Zbib, and D. Zych, "The iLab Shared Architecture: A Web Service Infrastructure to Build Communities of Internet Accessible Laboratories," *Proc. IEEE*, vol. 96, no. 6, pp. 931-950, 2008, doi:10.1109/JPROC.2008.921607.

Mario A. Bochicchio received the Dr. degree in "Ingegneria Elettronica" and the PhD degree in "Ingegneria Informatica" from the Technical University of Bari in 1990 and 1995, respectively. Since 1997, he has been at the University of Lecce, first as a researcher, and then, as an associate professor since 2001. He is currently an associate professor of software engineering and information systems at the University of Salento and the DIDA coordinator (http://dida.unile.it). His research interests include the areas of conceptual modeling of software systems and new technologies for teaching and learning. He has been involved in two European projects, seven national research projects, and a number of regional projects about the conceptual design of Web systems. Since 2006, he has been developing a collaborative Web system for small-medium groups of students (up to 30 people) to remotely controlled laboratory equipment via ADSL. He is a member of the IEEE and the IEEE Computer Society.

Antonella Longo received the master's degree and the PhD degree in information engineering from the University of Salento. She is currently an assistant professor at the University of Salento, where she teaches databases with the engineering faculty. She has been a consultant to the Italian Education Ministry on the promotion of scientific and technological culture. Her research interests include the design of services with particular interest for the learning environments. She is a member of the IEEE and the IEEE Computer Society.