Int. J. Engng Ed. Vol. 22, No. 4, pp. 849–857, 2006
Printed in Great Britain.

0949-149X/91 $3.00+0.00

Remote Lab Experiments Models: A Comparative Study

IMAD MOUGHARBEL
Electrical and Computer Engineering Department, Lebanese University, P.O. Box: 17-5091, Beirut, Lebanon

ALI EL HAJJ, HASSAN ARTAIL and CHADI RIMAN
Electrical and Computer Engineering Department, American University of Beirut, P.O. Box: 11-0236, Beirut, Lebanon. Email: elhajj@aub.edu.lb

Remote Laboratory Experimentation (RLE) is a technique used in modern engineering laboratories to help academic researchers and students perform laboratory experiments remotely through the Internet. Many RLE implementations are available with different characteristics. In this work, some recent RLE implementations models are analyzed, the services provided by each model are discussed, and these models are compared and evaluated.

INTRODUCTION

REMOTE LABORATORY EXPERIMENTATION (RLE) is a technique used in modern engineering laboratories to help academic researchers and students perform laboratory experiments remotely through the Internet [1, 2]. A typical RLE environment is shown in Fig. 1. From the client side, a computer is connected to the Internet with a web browser through which the experiment is conducted. On the server side, there are two important components: a lab server and a web server. The former consists of a computer connected to the experiment's hardware and possibly to a webcam. The web server, which is connected to the lab server, is responsible for managing the access by clients to the experimental setup. RLE offers the following advantages:

- Eliminates the need for physical presence in the lab and thus reduces the cost of running the experiment.
- Represents a form of ubiquitous computing (not confined to space/time).
- Allows accessibility for the disabled.
- Promotes student self-learning when used for education.
- Allows for time sharing of experimental hardware.
- Provides for resource sharing by organizations for unique or expensive equipment.
- Provides scalability in terms of user sessions and experiment size.
- Permits for storing experiment data and playing back the results to perform inspection analysis.
- Allows for demonstrating real experiments in real time through seminars and presentations.

* Accepted 16 February 2005.

Many RLE implementation models exist with different characteristics, and using different technologies. Due to the lack of well specified requirements for RLE, these different models do not provide a consistent set of services [3–17]. That is, some basic services are either different from one model to another, missing from particular models, or do not offer the exact needs. In this work, some existing models will be analyzed, evaluated, and compared. However, a detailed evaluation of the experimental environment is not included due to the large number of details that are specific to different experiments. This work therefore considers the technologies used and the services offered by RLE models.

IMPLEMENTATION TECHNOLOGY

In this section, the different technologies and tools used in the implementation of the surveyed models are presented. In many cases, different tools are used at the client side, server sides, and the experiment side. In addition to standard tools such as databases, web browsers, and programming languages the following tools are used:

- **LabVIEW:** The Laboratory Virtual Instrument Engineering Workbench (LabVIEW) software is developed by *National Instruments* to control the experiment and collect results. In many recent experimental setups, It is becoming the tool of choice for controlling the experiment, and offers drivers that allow it to control a large number of well-known instruments [10]. LabVIEW includes a listener program 'Courier New' that accepts parameters provided by the interface software, supplies them to the lab instruments, collects data from the instruments, and sends it

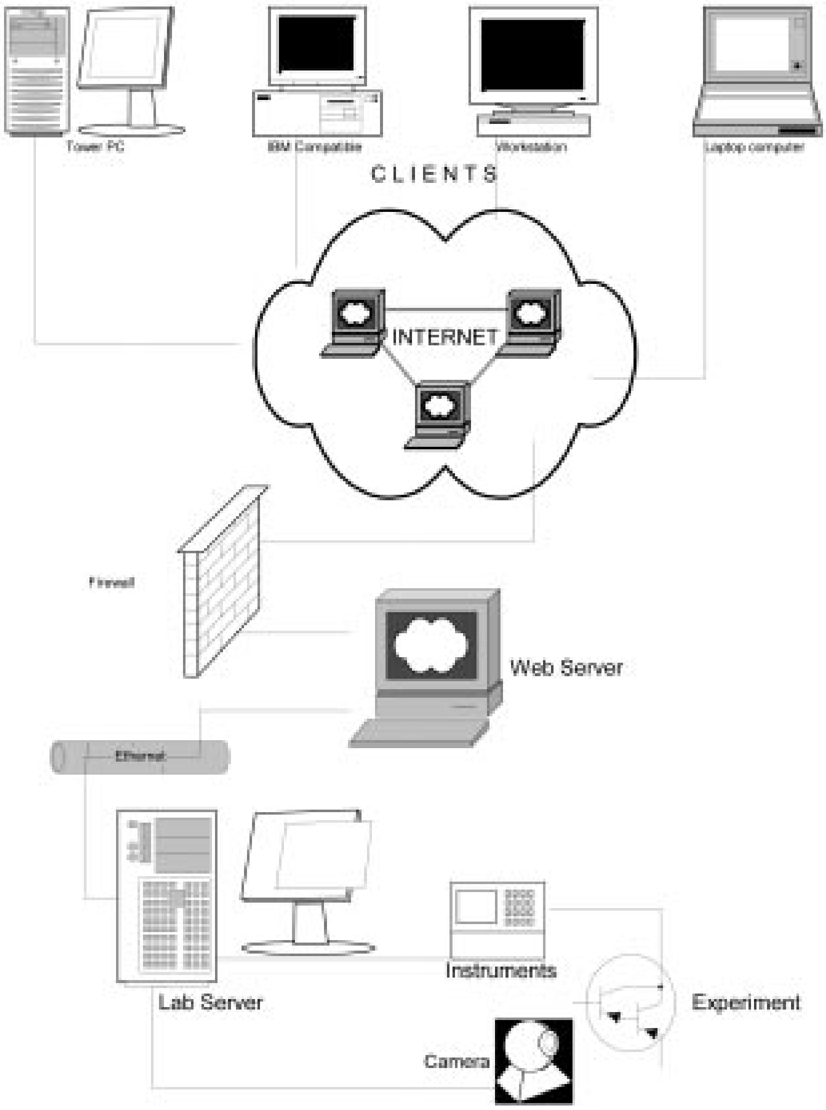

Fig. 1. A remote lab experiment environment.

back to the interface software. The user sends command strings to the lab instruments through the LabVIEW environment that routes the commands to corresponding ports (GPIB or others). The instruments interpret the messages and take actions accordingly. The user can read results (if any) from the Data Acquisition (DAQ) board that interfaces to the instruments. LabVIEW offers an integrated web server that can be used to publish a LabVIEW experiment on a web page.

- **VISA:** The Virtual Instrument Software Architecture (VISA) is an application programming interface (API) that facilitates instrumentation programming. It is a high-level driver that calls the lower-level drivers for each instrument and makes it easy to configure and control serial, GPIB, and VXI instruments. VISA achieves Interface independence by using the same methods to communicate with instruments, regardless of the interface type. For example, the VISA command to write an ASCII string to a message-based instrument is the same for serial, GPIB, and VXI. VISA has an object-oriented architecture that can easily accommodate new instrumentation interfaces.
- **CORBA:** The Common Object Request Broker Architecture (CORBA) specification standard for object-oriented middleware. It enables client objects to request operation executions from distributed server objects. It provides facilities and services that can be used to facilitate local and remote control of experiments.
- **COM+:** The Component Object Model+ (COM+) is an object-oriented middleware similar to CORBA but is implemented by Microsoft Corporation in contrast to CORBA, which is developed and maintained by the Object Management Group (OMG). COM+ has distribution capabilities which can also be used for local and remote control of experiments.
- **XML, XHTML, and JavaScript:** The Extensible Markup Language (XML) enables document authors to create their own markup for virtually

any type of information. XHTML, the Extensible Hypertext Markup Language specifies the format of text that is displayed in a web browser. JavaScript is a powerful scripting language that facilitates designing computer programs for enhancing the functionality and appearance of web pages.

- **Java Applets:** Java applets are client-side Java programs that maybe embedded in web pages and execute in a web browser.
- **CGI/Perl:** Practical Extraction and Report Language (Perl) is a widely used language for web programming. The Common Gateway Interface (CGI) is a standard protocol through which users interact with applications on web servers. CGI provides a way for clients (web browsers) to interface indirectly with applications on the web server. Because CGI is an interface, it cannot be programmed directly and hence, a script or executable program (called a CGI script) must be executed to interact with it. Perl is commonly used for programming CGI scripts due to its power and flexibility.
- **JSP:** JavaServer Pages (JSP) provide a way to build server-side web-based applications. JSP enables web-application programmers to create dynamic web content using XML syntax and scripting.
- **ASP/ASP.NET:** Microsoft's Active Server Pages (ASP) represent a server-side scripting technology that dynamically builds documents (e.g. XHTML, and XML) in response to client requests. Active Server Pages use server and client information to create dynamic web pages and send them to clients. ASP.NET is a combination of web Forms and web Services that build on Microsoft's web development technologies. Using ASP.NET, it is much easier to create dynamic and data-driven web applications that work well across browsers without any custom coding.
- **Java Media Framework (JMF):** The Java Media Framework (JMF) is an application programming interface (API) for incorporating media data (audio and video) into Java applications and applets. JMF has playback capability, in addition to capture, transmission, and transcode (conversion between file formats) functionalities.
- **Microsoft DirectShow:** Microsoft DirecShow & trade is a media-streaming architecture for the Microsoft Windows platform that enables capture and playback of multimedia streams. It supports video and audio data compressed in a wide variety of formats, including MPEG, Apple QuickTime, audio-video interleaved (AVI), and WAV files.
- **NetMeeting:** NetMeeting is Microsoft's real-time communication client that includes support for conferencing and allows two or more people to share (view and control) any Windows-based program across the Internet, a local area network, or the public telephone network. NetMeeting's Whiteboard, Chat, file transfer, and shared-clipboard tools allow groups of people to share information, conduct meetings, and jointly annotate diagrams and text in a shared workspace.
- **ConferenceXP:** Conference Experienced Project (ConferenceXP) is a software built by Microsoft to connect multiple distant participants for distance conferencing, instruction and collaboration. It allows real-time audio and video interaction, video streaming, and text-based collaboration through tools such as instant messaging, online chat, and discussion forums. ConferenceXP's architecture provides a flexible infrastructure for high-bandwidth interpersonal interaction as evidenced by its integration into instruction in several universities.

EVALUATION OF EXISTING MODELS

Many RLE models exist with characteristics that are similar in certain cases but still differ in one or more aspects. As it is not possible in a single paper to consider all existing models, sixteen models with different characteristics were selected and summarized. This paper describes the technology used in their implementation, identifies their services, and evaluates their characteristics. The surveyed models cover a wide range of topics such as computer engineering, electronics, mechatronics, chemistry and others. They are used for teaching and/or research and they provide services such as concurrency, collaboration, authentication, video streaming, data archiving, scheduling, and others. Each of these services however has varying characteristics in the different models. This diversity can be attributed to the lack of studies on the requirement specifications and standardization of RLE models and systems.

Collaboration, as it relates to this study, is related to a group of many students at different locations who are conducting the same experiment. This can be facilitated by existing groupware and collaborative technologies such as Netmeeting and Lotus Notes, in addition to customized tools for sharing access to an instrument or tool. The following are important characteristics that represent key components for determining the effectiveness of a particular model:

- Concurrency is for allowing different students to simultaneously access an experiment in read and/or write mode.
- Video streaming is employed to add the feeling of physical presence at the experiment's site through the employment of a video camera.
- Data archiving is used to make results from experiments persistent and available for later usage by students and instructors.
- Scheduling is implemented to facilitate access to the lab experiments without conflicts and congestion.

- Authentication is used in order to restrict access to the lab experiments to authorized users only.

The models that are studied will be enumerated by first, second . . . all the way to fifteenth, and are illustrated in the next subsections.

First model

The main objective of this model [3] is to include collaboration in an existing remote lab experimentation where collaboration and video streaming are achieved using ConferenceXP. The other offered services include user authentication, and a multi-read/one-write concurrency scheme. A web-based application is also developed for the user interface making it possible for the instructor to make lab assignments available online by activating web access to the lab resources. The model does not support a multiple read/write concurrency scheme, and does not provide means for experimental data archiving, and requires a high Internet bandwidth to achieve collaboration.

The model is applied for a computer network instructional course and provides the instructor the flexibility to insert or modify the laboratory material. This laboratory is a powerful educational and guiding environment for the students as it provides many important on-line features such as description of experiments objectives, directions and guidance, list of existing resources, discussion with the instructor, slides presentation, diagnosis for proper configuration and indicators for improper students' configuration.

Second model

This model is designed to build a mechatronics laboratory to observe mechanical vibration in machines and equipment [4]. The local and remote control, in addition to the web server, are implemented using LabVIEW. The employed client software is the LabVIEW run-time engine (also called LabVIEW Player). The services given by this model include authentication, and data archiving while lacking support for scheduling, video streaming, concurrency, collaboration and system management.

An example of an experiment would allow students to design a program that gets transmitted to the server for running the intelligent vibration monitor system. This program is first tested locally to ensure that it operates properly, and then it is tested remotely in monitoring and learning modes. The remote interaction is restricted to some specified operations which limit the distance learning features of the system.

Third model

The main objective of this model [5] is to implement a powerful tool for remote collaboration in experiments or observations requiring one or more facilities. It is based on an object-oriented distributed system called DYNACORE (DYNAmically COnfigurable Remote Experiment monitoring and control), which is developed using OMG's CORBA as the infrastructure for distributed computing, and in which the web server is implemented using C++. The client software is based on Java applets and a standard Internet browser. A generic instrument model is considered in the design and is defined in terms of a list of commands and events plus a graphical state diagram.

The strongest feature of this system is its collaboration capabilities through which users know the actions executed by their colleagues, and a chat server that allows live discussions using videoconferencing. Other services are also provided such as scheduling, video streaming and administration services, which include user management, schedule management and system configuration. Lastly, it should be noted that this model does not support data archiving.

Fourth model

This model [6] uses LabVIEW for direct control of the experiment and IBM VisualAge for remote control. Video streaming is achieved using ActiveX control in the LabVIEW environment. The web application is developed using Java Applets to generate the client control, and Java Server Pages (JSP) to generate the server-side scripts. Collaboration, concurrency, and data archiving are capabilities that are not supported. The model illustrates the use of some implementation techniques on a simple experiment, but can be used to develop more sophisticated experiments.

Fifth model

This model's title is AIM-Lab (Automatic Internet Measurement Laboratory) [7]. It is developed using Microsoft Visual C++ to generate the server side, and Java Applets for the client control. The server part includes a TCP/IP server socket that receives commands from the Internet, a driver interface layer (DIL) that interfaces between the instrument driver and the application layer of the server, and a graphical user interface (GUI) which allows the instructor to monitor and control the server process and modify the configuration of the instrumentation. The client sends commands to the server through the Internet, and uses a command generator that issues commands according to the parameter set that are specified by the user. There is no support for authentication, scheduling, collaboration, concurrency, data archiving and video streaming.

The model was tested in the university campus and students' comments and critique helped in improving the design for other remote laboratories. Students were able to remotely conduct experiments related to electronics components characterization. Student satisfaction was observed and attributed to the fast response time and the possibility of launching measurements consecutively without waiting for results of previous ones.

Sixth model

This model (LAB-on-WEB) was developed at UniK University Graduate Center near Oslo, Norway [7]. It uses Microsoft COM+ and LabVIEW. The LabVIEW Internet Developer Kit with CGI scripts is used at the server side while the LabVIEW Player (client runtime engine), is used at the client side to run the experiment. This model does not support authentication, scheduling, collaboration, concurrency, data archiving or video streaming.

The main educational objective of this model is to let the student perform individual discovery of the material using a remote electronic devices laboratory. This model was successfully used by students from different campuses who are enrolled in a pilot course that was part of a distance learning program.

Seventh model

This model is the MIT Microelectronics webLab [8] where the purpose is to make microelectronics device characterization over the Internet possible for students to be able to take current-voltage measurements on transistors and other devices from anywhere and at anytime. webLab has two architectures, one for single-device experimentation and one for multiple devices. Both architectures essentially share the same components.

On the client side, Java applets provide the user interface to the experiment and validate test requests to the webLab server. On the server side, Microsoft Internet Information Services (IIS) receives the client request and interacts with the client through a set of Active Server Pages (ASP). IIS includes a request queue to queue all pending client requests before being processed. ASP pages send client requests to webLab driver, programmed in Visual Basic, which in turn sends commands to the VISA driver for translating the Visual Basic commands to GPIB commands that are directly understood by the Lab equipment. Finally, an SQL Server database records and supplies user information by communicating with the ASP engine. The services given by this model include data archiving, authentication and system management pages for administrators to control the experiments but does not support scheduling, collaboration, concurrency, video streaming, and interaction with the instructor.

Eighth model

This model's title is RETWINE (Remote Worldwide Instrumentation Network) [9] and was designed for the goal of allowing the use of powerful instruments via the World-Wide Web. The overall system architecture consists of measurement instruments, an instrumentation server, a video server, a web server and client computers. A controller program implemented in the C language runs on the instrument server to manage the data exchange between the web server and the instruments (through GPIB interfaces). The web server is used to manage user access control to the experiments. It gives information about available measurement instruments, authenticates users, and checks for reservations. Further, it acts as a communication bridge for the information flow between the user and the measurement system. It also provides screenshots and live pictures of the instruments using webcams that are connected to the video server (VS).

On the client side, Java applets provide extra capabilities to the user interface while NetMeeting allows for the communication between the user and the operator in charge of the equipment. This model supports authentication, scheduling, video streaming, and collaboration between students and instructors but does not support data archiving, concurrency on same equipment, and collaboration among users.

It was used in teaching students how to use real measurement instrumentation. A drawing of the instrument's front panel is transmitted to the student who is able to control remotely the real instrument by mouse clicks on the front panel. Students and instructors satisfaction was noticed and a need for improved communication was suggested while noting that some instructors encountered difficulties in experiment configuration.

Ninth model

This model is developed as part of the PEARL project [11] and used in four universities: Open University, Trinity College (Dublin), University of Porto, and University of Dundee. The experiments developed are: Motorized Optical Spectrometer Jig, Computer Vision Experiment, and test of digital and mixed-signal circuits. The local control of the experiments is done via programs in C++, Java, and Java JMF. The remote experiment's controls use CORBA and the web server depends on servlets to invoke the CORBA interface, while the client's interface is based on Java applets. The services provided by this model include authentication, video streaming, collaboration, and concurrency but does not include scheduling, data archiving, and system management. The system was revised several times and has become more user friendly but still suffers from response delays, which are likely to be solved in future revisions.

This model was employed in implementing three different experiments. The first experiment introduces students to the optical spectra for different chemicals, the second teaches students about basic camera optics and image processing algorithms and the third deals with testing digital and mixed signal circuits. A structured questionnaire has shown that the collaborative aspect of the model has considerably improved the educational value while it was noticed that communication delays could disorient and frustrate students.

Tenth model

This model describes the remote laboratory used in the University of Tennessee at Chattanooga (UTC) [12]. The local and remote control of the experiments is done via LabVIEW and the client's interface is built on top of LabVIEW player. The model supports authentication, scheduling, data archiving, collaboration and concurrency but lacks the support of video streaming and system management.

This model includes different techniques for implementing three different types of experiments: mechanical, control and chemical. The mechanical experiments are on kinematics of motion, linear vibrations and heat exchange. The control experiments involve remotely operable control stations based on control elements such as: speed, voltage, level, etc. The chemical experiments are on packed column absorption, distillation, heat exchange, flow through porous media and so on.

Eleventh model

This model is the TIPY platform developed and used in the ICTT laboratory of the University of Lyon, France. The experiment developed is a vertical store driven by two programmable logic controllers (PLCs) and used by industrial engineering students [13]. In order to increase the educational value, the system allows for comparing simulation results to real experimental data. During experimentation students learn how to program a PLC after learning how to implement a system functional model. Finally, they remotely execute programs that they have written to control a vertical storage system.

The local and remote control of the experiments is programmed using ladder logic, which the web server is based on PHP pages that are run by an APACHE server which is connected to a MySQL database, and finally, the client's interface is implemented using Java applets. The services given by this model include authentication, scheduling, video streaming, concurrency, and system management but lacks collaboration and data archiving. The user friendliness of the system can benefit from communication tools such as videoconferencing and virtual whiteboards. Technical stability is not fully reached since the software does not run on multiple platforms.

Twelfth model

This model is the TORUS (Toys Operated Remotely for Understanding Science) project built to use toys like diggers, cranes and bulldozers, to create simple task scenarios [14]. The local and remote control is implemented using multithreaded C programming. Telnet is used instead of a web server while the client software was implemented using Delphi. The services given by this model include authentication, video streaming, and concurrency but are missing scheduling, collaboration, data archiving and system management.

The applications of this model have to do with the idea of placing toys in appropriate scenarios so they can be manipulated in robotics and artificial intelligence. Students are often challenged to add a certain level of intelligence to the toy devices so as to make the experiments more dynamic. Through an evaluation a high degree of student satisfaction was noted, which was indicative of the knowledge acquired, although some difficulties were encountered in complementing the assignments.

Thirteenth model

This model is used by students to tele-operate a robot in a laboratory [15]. The local control is implemented with the SCORBASE language while the web server application, the remote control, and the client's software were implemented using Java, Java sockets and Java applets respectively. The services given by this model include authentication, scheduling, video streaming, and data archiving but exclude concurrency, collaboration and system management. The model is technically stable whereby a double validation is performed for commands that get sent to the web server in order to prevent robot system's damage.

The robotic application considered in this model, requires a fast response time when operating in a particular mode. Instead of using an expensive communication technique, a virtual method was used to satisfy this requirement and to let students feel that they are close to the robot. In fact, the real robot parameters are transmitted via the Internet to a virtual reality interface that shows graphically the robot status.

Fourteenth model

This model is the RoboWEB lab used in the mechatronics course at the University of Cagliari, Italy. It allows students to remotely access LEGO programmable robots [16]. The instructor manages the robot's operation to be performed by students and validates any code to be downloaded. The local and remote control are implemented using the RoboLAB software which has many features that are similar to LabVIEW, and the web server and client software were developed using standard HTML. The services given by this model include authentication, and video streaming and are missing scheduling, data archiving, concurrency, collaboration and system management pages.

Fifteenth model

This model is called RELATED (Remote Laboratory Extended) and is an XML-based framework for the development of Internet-based laboratory systems [17]. An implementation of this model is used in a small-scale pasteurization plant which is an energy exchange between two liquid flows. The local control is implemented using a Windows application called CILab while the remote control is developed using Java. XML code is mainly used for the web server and for configuring remote laboratory experiments. As in many other systems, the client software is

Table 1. Characteristics of different models.

Model	Area	Student level	Purpose
1	Computer Networks	Senior	Research/Teaching
2	Mechatronics	Graduate	Research
3	Physics	Unrestricted	Teaching/Research
4	Electronics	Unrestricted	Teaching
5	Electronics	Graduate	Teaching
6	Electronics	Senior	Teaching
7	Electronics	Senior/Graduate	Teaching
8	Instrumentation	Senior	Teaching
9	Engineering and Science	Senior/Graduate	Teaching
10	Mechanical/Control/Chemical	Unrestricted	Teaching
11	Industrial Engineering	Senior	Teaching
12	Robotics	Senior	Teaching
13	Robotics	Unrestricted	Teaching
14	Mechatronics	Senior/Graduate	Teaching
15	Heat transfer	Unrestricted	Research/Teaching

implemented using Java applets. The services offered by this model include authentication and data archiving but lack scheduling, video streaming, concurrency, collaboration and system management.

COMPARISON OF MODELS

In this section, the fifteen models discussed in the previous section are compared and the comparison is summarized in Table 1, which shows the characteristics of different models. Table 2 highlights the technologies used while Table 3 goes over the services provided. Table 1 shows that RLE is used in many areas, namely mechanics, chemistry, control, electronics, and others. It is used in universities for teaching and research, and for different levels.

Table 2 shows that different technologies are used in many implementations of the RLE. LabVIEW is the most widely used tool for local and remote control of experiments. However, in some cases when no drivers for LabVIEW are available, some programming is necessary using languages such as C, C++, Visual Basic and others. The web server is implemented in many cases using ASP or CGI and powered by Javascript, VBscript or PHP. Java applets are used in many cases at the client side, and LabVIEW player can also be used when LabVIEW is used for the remote control of the experiment.

Finally, Table 3 shows that different models support different services. For example, most models do not support data archiving, collaboration and concurrency in spite of their importance. This is due to the lack of a global abstract model or guidelines that are followed to implement an RLE. It is concluded that an analysis study is needed to specify the services and their specific requirements in order to reach a standardized model that could be used for the implementation of future RLEs.

Additionally, other issues were reported with most of the models relating to delay and lack of

Table 2. Technologies used by different models.

Model	Local control of experiment	Remote control of experiment	Web server	Client
1	R-Lab	R-Lab	Active Server Pages (ASP) + ConferenceXP	Java Applets + ConferenceXP
2	LabVIEW	LabVIEW	LabVIEW	LabVIEW run-time
3	CORBA	CORBA	C++ agents & modules	Java Applets
4	LabVIEW	IBM VisualAge	Java Server Pages (JSP)	Java Applets
5	Custom program using Visual C++	Custom program using Visual C++	Windows-based Multidocument Interface	Java Applets
6	COM+ LabVIEW	COM+ or LabVIEW	ASP or CGI	HTML + Java Script + XML + SVG or LabVIEW Player
7	VISA driver	Visual Basic	Active Server Pages	Java Applets
8	RPC using C	Custom program using C	Multithreaded daemon using Java	Java Applets + NetMeeting
9	C++, Java, Java JMF	CORBA	Servlets, CORBA	Java Applets
10	LabVIEW	LabVIEW	LabVIEW	LabVIEW player
11	Special program. env.	Special program. env.	PHP	Java Applets
12	Multi-threaded C	Multi-threaded C	–	Delphi
13	SCORBASE	Java sockets	Java	Java Applets
14	RoboLAB	RoboLAB	HTML	HTML
15	CILab	Java	XML	Java Applets

Table 3. Services provided by different models.

Model	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Authentication	X	X	X	X			X	X	X	X	X	X	X	X	X
Scheduling	X		X	X				X		X	X		X		
Video Streaming	X		X	X				X	X		X	X	X	X	
Data Archiving		X					X			X			X		X
Collaboration	X		X					X	X	X					
Concurrency*	mrow		mrow						X	X	X	X			
System Administration	X		X		X		X				X				
web system Administration			X				X								

* mrow: Multi Read, One Write

user friendliness (Model 9, for instance). The delay is sometimes due to the large files transmitted over the network and to the type of applications that require speed in data acquisition. In many cases, user friendliness and educational value were increased through student surveys which led to determining the type of needed improvement.

CONCLUSION

In this work, some recent RLE implementations models were analyzed and compared and the provided services were discussed. It was found that different models support different services and the design approach doesn't follow any standard model or guidelines. This diversity is due to the lack of studies on the requirement specifications and standardization of RLE models and systems. Some of the services identified were concurrency, collaboration, authentication, video streaming, data archiving, scheduling and administration. There is no evidence that these services are the only needed ones for an RLE setup. An analysis study is therefore needed to specify the services and their specific requirements in order to reach a common standard model that could be used to implement future RLE. This analysis should follow a suitable software engineering approach to determine a set of complete and correct requirement specifications [18].

Acknowledgment—This work was supported by the CEDRE Program.

REFERENCES

1. S. H. Chen, R. Chen, V. Ramakrishnan, S. Y. Hu, Y. Zhuang, C. C. Ko and B. M. Chen, Development of remote laboratory experimentation through Internet, *Proc. 1999 IEEE Hong Kong Symposium on Robotics and Control*, Hong Kong, July 1999, pp. 756–760.
2. A. El Hajj, I. Mougharbel and C. Riman, On the design and implementation of remote computer controlled laboratory experiments, *CIBITIC Conference*, May 2004, Beirut, Lebanon.
3. J. Hua and A. Ganz, A new model for remote laboratory education based on next generation interactive technologies—a generic laboratory plug-in using MS ConferenceXP learning infrastructure, *Proc. ASEE New England Regional Conference*, 2003, Orono, ME.
4. K. Tan, *et al.*, Mechatronics experiment on remote vibration monitoring and fault diagnosis via the intranet, *Int. J. Eng. Educ.* **19**(3), 2003, pp. 503–511.
5. J. A. Rodriguez, E. Garcia, J. Rejas and M. A. Duran, Web-based distributed systems for collaborative remote experiments, *Int. Conf. Accelerator and Large Experimental Physics Control Systems*, 1999, Trieste, Italy, pp. 483–485.
6. A. O. Abdul Salam and T. Ozkul, Java-based cyber technique for on-line engineering labs, *Proc. 2nd Int. Conf. Principles and Practice of Programming in Java*, 16–18 June, 2003, pp. 109–112.
7. T. A. Fjeldly and M. S. Shur, Electronics laboratory experiments accessible via Internet, in *Lab on the Web*, Wiley Interscience (2003) pp. 1–47.
8. J. A. Del Alamo, V. Chang, L. Brooks, C. McLean, J. Hardison, G. Mishuris and L. Hui, MIT Microelectronics webLab, in *Lab on the Web*, Wiley-Interscience (2003) pp. 49–87.
9. T. Zimmer *et al.*, Instrumentation on the web, *Lab on the Web*, Wiley-Interscience (2003) pp. 89–143.
10. M. Naghedolfeizi, S. Arora and S. Garcia, Survey of LabView technologies for building web/Internet-enabled experimental setups, *Proc. 2002 ASEE Annual Conference and Exposition*, June 16–19, 2002, Montreal, CA.
11. M. Cooper, A. Donnelly and J. Ferreira, Remote controlled experiments for teaching over the Internet: a comparison of approaches developed in the PEARL project, *ASCILITE 2002 Conf. Proc.*
12. J. Henry and C. Knight, Modern engineering laboratories at a distance, *Int. J. Eng. Educ.* **19**(3) 2003, pp. 403–408.
13. A. Leleve, H. Benmohamed, P. Prevot and C. Meyer, Remote laboratory towards an integrated training system, *ITHET'03*, Marrakech, MOROCCO, July 7–9, 2003.
14. G. McKee, An online robot system for projects in robot intelligence, *Int. J. Eng. Educ.*, **19**(3) 2003, pp. 356–362.
15. F. Candelas *et al.*, A virtual laboratory for teaching robotics, *Int. J. Eng. Educ.*, **19**(3) 2003, pp. 363–370.

16. A. Serri, A novel website of mechatronics for remote learning, *Int. J. Eng. Educ.*, **19**(3) 2003, pp. 420–426.
17. R. Pastor, *et al.*, A XML-based framework for the development of web-based laboratories focused on control systems education, *Int. J. Eng. Educ.*, **19**(3), 2003, pp. 445–454.
18. S. L. Pfleeger, *Software Engineering Theory and Practice*, 2nd Ed., Prentice-Hall (2001).

Imad Mougharbel was born in Beirut, Lebanon. He received the Engineering Diploma in Electronics from Ecole Centrale de Lyon, France in 1977, the degree of 'Docteur Ingenieur' from the University of Aix-Marseille, France, in 1980. In 1980, he joined the Lebanese University where he is currently Associate Professor in the Electrical and Electronic Department. From 1997 to 2004 he was senior lecturer at the American University of Beirut in the Electrical and Computer Engineering Department. From 1990 to 1994 he was Research Associate at the Electrical Department of the 'Ecole Polytechnique de Montréal' in Canada. His research interests are Electrical power conversion and Remote control systems.

Ali El-Hajj was born in Aramta, Lebanon. He received the Lisense degree in Physics from the Lebanese University, Lebanon in 1979, the degree of 'Ingenieur' from L'Ecole Superieure d'Electricite, France in 1981, and the 'Docteur Ingenieur' degree from the University of Rennes I, France in 1983. From 1983 to 1987, he was with the Electrical Engineering Department at the Lebanese University. In 1987, he joined the American University of Beirut where he is currently Professor of Electrical and Computer Engineering. His research interests are numerical solution of electromagnetic field problems and engineering education.

Hassan Artail Worked as a system development supervisor at the Scientific Labs of DaimlerChrysler, Michigan before joining AUB in 2001. At DaimlerChrysler, he worked for 11 years in the field of software and system development for vehicle testing applications, covering the areas of instrument control, computer networking, distributed computing, data acquisition, and data processing. He obtained a BS and MS in Electrical Engineering from the University of Detroit in 1985 and 1986 respectively and a Ph.D. from Wayne State University in 1999. His research is in the areas of Internet and Mobile Computing, Distributed Computing and Systems, and computer plus network security.

Chadi Riman was born in AinZhalta, Lebanon. He received his Bachelor of Engineering and Masters of Engineering degrees both in Computer & Communication Engineering from the American University of Beirut (AUB), Lebanon in 1994 and 2004, respectively. He is currently pursuing his PhD degree at the University of Versailles (UVSQ), France. He worked from 1994 to 1998 in the software engineering domain, and became the IT manager in AinWazein Hospital, Lebanon since 1999. His research interests include remote engineering education and software systems for handicap rehabilitation.