AC 2009-1350: TECHNOLOGIES FOR THE DEVELOPMENT OF VIRTUAL AND REMOTE LABORATORIES: A CASE STUDY

Xuemin Chen, Texas Southern University

Lei Jiang, Donghua University

Darayan Shahryar, Texas Southern University

Lawrence Kehinde, Texas Southern University

David Olowokere, Texas Southern University

Technologies for Development of Virtual and Remote Laboratories – A Case Study

Abstract

With the rapid development of computer and Internet technologies, the virtual and remote laboratories have become an important part of the educational process. To develop interactive virtual and remote laboratories (VR-Lab) for Engineering and Technology education, several tools are available. The most popular are LabVIEW, Java applet and Flash. In this paper, case studies are presented for the development of VR-Lab with LabVIEW, Java applet and Flash. The client software requirements comparison is drawn from virtual and remote laboratories development practices.

Introduction

From the earliest days of engineering education, hands-on laboratories have been an essential part of undergraduate engineering programs[1]. Concepts taught through lectures are often complemented with hands-on experimentations. Hands-on experiments help students to understand the backbone of science and engineering by observing dynamic phenomena, testing hypotheses, learning from their mistakes, and reaching their own conclusions.

With the rapid development of computer and Internet technologies, computer and Internet based learning has become an important part of education. The Sloan Survey of Online Learning, "Staying the Course: Online Education in the United States, 2008", shows that over 3.9 million students were taking at least one online course during the 2007 Fall term; a 12 percent increase over the number reported the previous year[2].

Providing theoretical educational materials online is a relatively simple task, where several multimedia tools and editors such as Hyper Text Markup Language (HTML), and Extensible Markup Language (XML) are available which can be used to create courseware[3, 4]. Developing educational visualization and simulation environments online are harder, but they are worth the effort because they support self-driven learning[5]. With the ever-increasing pace of technology renewal, it becomes simpler to place ''real'' laboratory learning environments online. New technologies allow students to conduct the laboratory simulation, the automated data acquisition and the remote control of instruments, all online. Currently, there are two approaches to conducting laboratories online, namely the Virtual and Remote laboratories.

Virtual laboratory is based on software such as National Instruments' Laboratory Virtual Instrument Engineering Workbench (LabVIEW), Java applet, Flash or other software to simulate the lab environment. A frequency modulation virtual laboratory[6] was developed using Java and LabVIEW. Additionally, a virtual microscopy was developed with Flash at University of Delaware[7].

Virtual labs can be used for experiments that would normally require equipment that is too expensive, unsafe (e.g. nuclear reactor) or unavailable. Virtual labs also allow students to repeat an experiment multiple times, giving them the opportunity to see how changed parameters affect the outcome. One of the very important features of the virtual lab is to let the students learn from failures without causing any real damages. Learning from failure is one of the nine objectives for the engineering education laboratory defined by ABET[8]. The "Virtual Reality Laboratory Accidents" project[9] was developed at University of Illinois Chicago by using virtual reality technologies such as Virtual Reality Modeling Language (VRML) and Java 3D. It is believed that these accidents will have more impact on users than written rules, even if not as much as real accidents.

Remote lab by definition is the experiment which is conducted and controlled remotely through the Internet. The experiments use real components or instrumentation at a different location from where they are controlled or conducted. For example, University of Houston offers access to their remote laboratory for Smart Materials[10]. The interactive mode for remote laboratory experiments has also been implemented in other remote lab environments. One of the well developed remote labs is the Massachusetts Institute of Technology (MIT) iLab project[11]. iLab relies on a three-tier Web architecture including client applications, service broker and lab server[12]. Java or LabVIEW was used to develop most of the laboratories in the iLab project.

To develop interactive virtual and remote laboratories (VR-Lab) for Engineering and Technology education, several tools are available. Based on the aforementioned review for virtual and remote laboratory development, the popular tools are LabVIEW, Java applet and Flash. In this paper, case studies are presented for development of VR-Lab with LabVIEW, Java Applet and Flash. The client software requirements comparison is drawn from virtual and remote laboratories development practices.

Architecture of VR-Lab

The system block diagram of the VR-Lab which is under development in the authors' department is shown in Figure 1. The functionality of the server is to work as the web publisher, the lab scheduler, as well as the data and database manger. The workstations are used to execute the users' requirements and control the lab devices such as the NI Educational Laboratory Virtual Instrumentation Suite (NI-ELVIS) and control plan to conduct the experiments. The camera will let the users see the system response in real time. The users can then use the client computer to do the experiments in virtual and remote way.

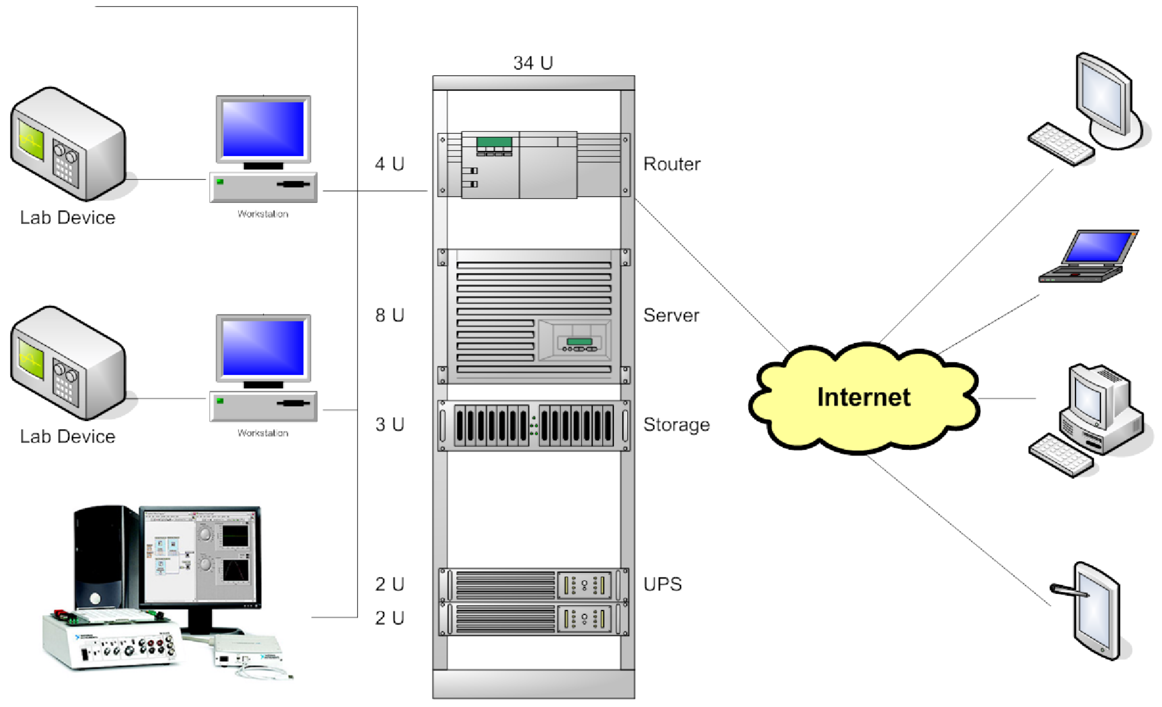
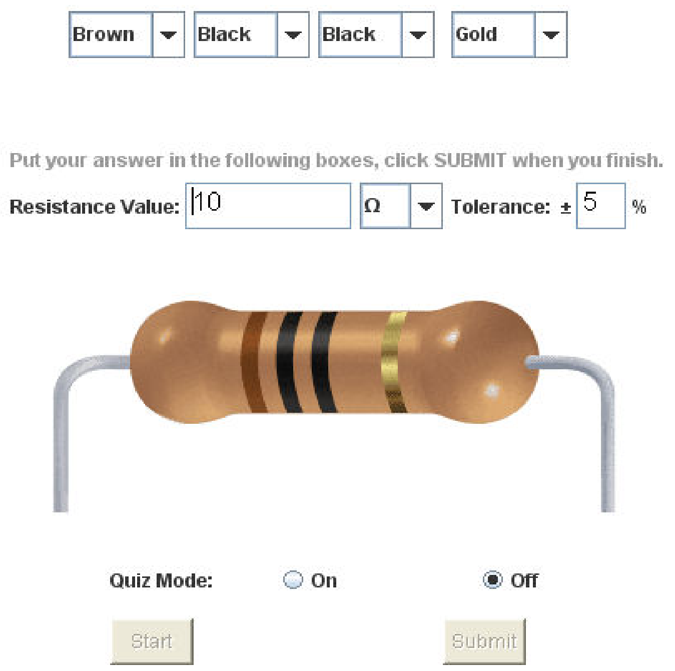


Figure 1. System Block Diagram of the VR-Lab

Resistor Color Code – A Virtual Laboratory Developed with Java Applet

Java was created by James Gosling at Sun Microsystems and released in 1995 as a core component of the Sun Microsystems' Java platform[8]. It promised "Write Once, Run Anywhere" (WORA), providing no-cost run-times on popular platforms. Fairly secure and featuring configurable security, it allowed network- and file-access restrictions. Major web browsers soon incorporated the ability to run secure Java applets within web pages, and Java quickly became popular[13]. Java has been involved in many virtual laboratories since then.

The first virtual laboratory we developed with Java Applet is the Resistor Color Code. It has two modes: the learn mode and the quiz mode. The default is the learn mode shown in Figure 2. In this mode, the user can use the combo box to select the different combinations of the color bands. The resistor value is then calculated by Java Applet. When the user picks the quiz mode, the Java Applet randomly generates a combination of color bands, after which the user inputs the resistor value into the textbox. A SUBMIT button is built for users submitting the answer for checking. A popup window will display a message such as “Correct” or “Wrong” shown in Figure 3.



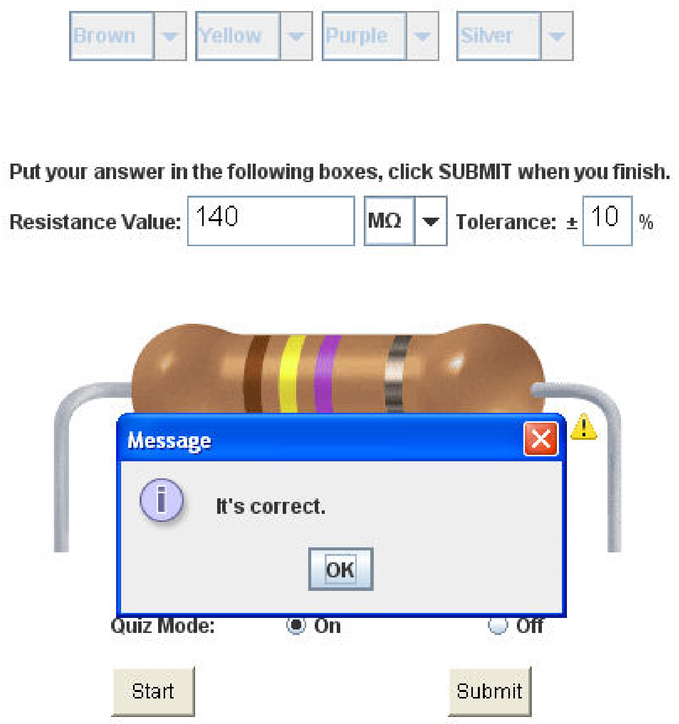
Brown | Black | Black | Gold

Put your answer in the following boxes, click SUBMIT when you finish.

Resistance Value: 10 Ω Tolerance: ± 5 %

Quiz Mode: [ ] On [● Off]

Figure 2. The default mode is quiz off. Resistor value is recalculated as the bands change.



Brown | Yellow | Purple | Silver

Put your answer in the following boxes, click SUBMIT when you finish.

Resistance Value: 140 MΩ Tolerance: ± 10 %

Message

It's correct.

Quiz Mode: [● On] [ ] Off

Figure 3. The quiz mode is on. The user reads the color code, fills in the box, and then submits their answer for checking.

In the resistor color code experiment, the resistor is composed by combining 7 images together. When user selects the color band, the corresponding color band image will be loaded. One of the most important features of Java is its support for images and sound. Knowing this, the Java architects have had to deal with a common problem associated with distributed media: transmission delay due to the limitation of Internet bandwidth. The problem which Java applets face in this case is that the images may not load properly. On a fast network connection, the images will load quickly, but on a slow modem or network connection, images would take longer, or may not arrive at all. Users could be confused by blank images - particularly when a sequence of images is being displayed. The Java media tracker (MediaTracker) is a Java object that helps deal with the transmission delay problem by keeping up whether media objects have been successfully transmitted. Of course, the media tracker is not the only way of handling the issue, but it is the simplest. The code shown in Figure 4 has been developed in the implementation of the Resistor Color Code virtual laboratory.

```
URL base;
MediaTracker mt;

public void init()
{
    .
    .
    .

    mt = new MediaTracker(this);
    try
    {
        base = getDocumentBase();
    }
    catch (Exception e) {}

    resLeft = getImage(base,"resleft.gif");
    mt.addImage(resLeft,1);
    resImages[0] = getImage(base,"rBrown.gif");
    mt.addImage(resImages[0],2);
    resImages[1] = getImage(base,"rBlack.gif");
    mt.addImage(resImages[1],3);
    resImages[2] = getImage(base,"rBlack.gif");
    mt.addImage(resImages[2],4);
    spacer = getImage(base,"rNone.gif");
    mt.addImage(spacer,5);
    resImages[3] = getImage(base,"rGold.gif");
    mt.addImage(resImages[3],6);
    resRight = getImage(base,"resright.gif");
    mt.addImage(resRight,7);

    try
    {
        mt.waitForAll();
    }
    catch (InterruptedException  e) {}
    .
    .
    .
    }
```

Figure 4. Using MediaTracker to load resistor images.

This Java applet (Resistor Color Code Virtual Laboratory) has been used to teach undergraduate students and high school students in a pre-college summer engineering program. The students can master this skill easily with a developed vivid user interface and quiz. This applet is being maintained as a sub-page of our department's website, http://engineering.tsu.edu. It is worth mentioning that our students have shown great interest in this virtual lab design which was assigned as a final project in the Java programming course. A well designed final project with a graphic user interface will motivate students to learn and better understand Java. The Java teaching methodology was published in the ASEE 2008 Annual Conference and Exposition[14]. Usually, virtual labs can be used to train students in the use of equipment prior to hands-on experiences.

Resistor Color Code – A Virtual Laboratory Developed with Flash

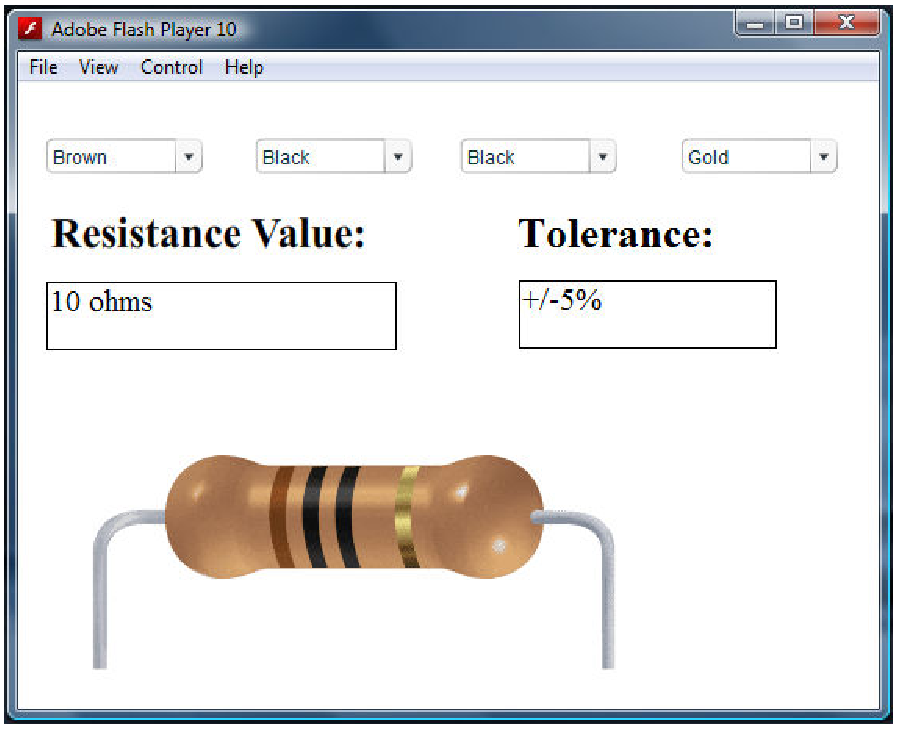
Flash is a multimedia platform created by Macromedia and released in 1996. Currently, it is developed and distributed by Adobe Systems[15]. Because of bandwidth concerns on the Internet, there has long been an urgent need for vector graphics. Flash is a vector animation software, originally designed to create lightweight animations on web pages. One advantage of Flash is that when a Flash application is built, only the graphics and additional media elements have to be stored. Their actions are defined by mathematical formulas instead of large data sets.

Another big advantage of Flash is that it has practically no browser issues. Since Flash files are only viewable with a plug-in, Flash movies will work the same if the user is on Firefox, Safari or IE, on Mac or PC. And the growth of the ActionScript programming language offers the chance to take it much farther, if desired.

Flash provides better copyright protection than Javascript does. The source code cannot be seen without using some special SWF decompilers.

Flash has become a popular method for adding animation and interactivity to web pages. There are few reports of using Flash for remote laboratory design. One of the Flash based remote laboratories was developed at the HAMK University of Applied Sciences, in Finland[16]. A Flash interface was developed for Programmable Logic Controller (PLC) control and real time PLC data display.

To explore the Flash virtual laboratory development in the VR-Lab framework, we developed the Resistor Color Code Calculator again with Flash 8 and ActionScript 2.0. A screenshot is shown in Figure 5.



Resistance Value:

10 ohms

Tolerance:

+/-5%

Figure 5. Resistor color code calculator developed with Flash.

Control Laboratory – A Remote Laboratory Developed with LabVIEW

LabVIEW is both powerful and flexible as a platform and as a development environment for the graphic programming language from National Instruments[11]. It is particularly suitable for developing Web-based virtual reality laboratories in scientific, engineering, and technological disciplines.

A sample experiment showing the front panel of LabVIEW for implementing a PID control of a second-order plant is shown in Figure 6. Students can vary PID gains as well as damping ratio, sampling time, and other system parameters to observe the response of the system. These experiments can be performed in a matter of minutes to verify the theory and concepts learned in a lecture course.

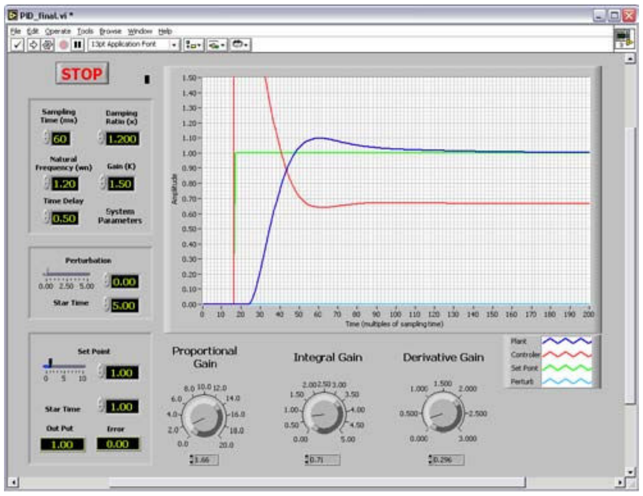


Figure 6. Front panel of PID control of a second-order plant using LabVIEW

Figure 7 shows a sample for process measurements and control. Complex systems that can be configured in different control modes will be implemented during the full implementation of this project. The proof-of-concept, as planned for this project, can be accomplished with a simple configuration shown in Figure 7.

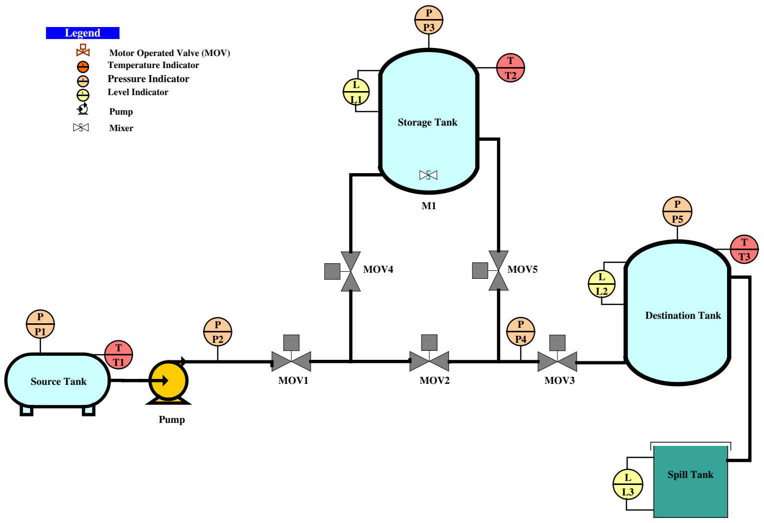


Figure 7. Typical Control System Plant

Client Software Requirements for VR-Lab

A useful feature included in the LabVIEW package is the “Internet Toolkit” which contains the G Web Server. The laboratory experiments based on the Virtual Instruments (VIs) concept can be easily made ready for Internet delivery with the use of LabVIEW’s Internet Toolkit. By enabling the G Web Server, the developer can easily publish the front panel of the application on the Internet. Then the images of the front panel of the published application can be accessed by students remotely. However, the client computers on which students want to run LabVIEW stand-alone applications or shared libraries must have the LabVIEW Runtime Engine installed. The client side Runtime Engine version depends on the application development version used by the developer. The versions are not interchangeable[16]. On the other hand, the LabView Run-Time Engine is quite large. The standard version of the LabView Run-Time Engine 8.5 is 97 MB. The web browser only version is 24 MB[17]. Another issue is that LabVIEW uses ActiveX to provide function/data/service sharing. Since ActiveX applies only to the Windows platform, the Windows Internet Explorer is required to browse the web interface developed by LabVIEW.

The JAVA Applet running on the student side sends a request for connection to the program running on the Web Server. The program then sends the connection request to the controller as well as accepts the request from the student. When the access is granted a link from the student to the controller is established to transfer commands and data. Since the Java Bytecode can only be executed by the Java Virtual Machine (JVM), the Java runtime support has to be installed also. But the JVM size is pretty small. It is only 7 MB for the latest Java update.

To work with the virtual laboratory developed with Flash, users need to download and install the Adobe Flash Player before they can view anything. The latest windows version plug-in is only1.8 MB for IE, Firefox, Safari and Opera.

The client software requirements comparison drawn from the virtual and remote laboratories development practices is shown in Table 1.

Table1. Comparison of Client Software Requirements

	Graphic Programming	Programming Complexity	Web Browser	Plug-in	Plug-in Size
LabVIEW	Yes	Medium	IE	Yes	Large
Java applet	No	High	Browser Independent	Yes	Medium
Flash	No	Low - High	Browser Independent	Yes	Small

Conclusions

In this paper, case studies are presented for the development of virtual and remote laboratories with LabVIEW, Java Applet and Flash. LabView is graphic programming language. The laboratory experiments based on the VI concept can be easily made ready for Internet delivery. A LabView Run-Time Engine must be installed on client side, but it has compatibility issues between the different versions. Java applet and Flash are becoming more popular. Most of the PC distributors have preinstalled for the users, but Object Orientated programming skills are required for the virtual and remote laboratory development.

Bibliography

1. L. Feisel, A. Rosa, "The Role of the Laboratory in Undergraduate Engineering Education", Journal of Engineering Education, pp. 121-130, January 2005.
2. Sloan Consortium of Institution and Organizations Committed to Quality Online Education, "Staying the Course: Online Education in the United States, 2008". http://www.sloan-c.org/publications/survey/staying_course
3. B. Balamuralithars, and P. C. Woods, "Virtual Laboratories in Engineering Education: The Simulation Lab and Remoter Lab", Computer Applications in Engineering Education, Vol 17, Issue 1, 2008, pp. 108-118.
4. H. J. W. Spoedler, Virtual instruments and virtual environments, IEEE Instrum Meas Mag 2 (1999), 14-19.
5. N. Jensen, G. von Voigt, W. Nejdl and S. Olbrich, "Development of a Virtual Laboratory System For Science Education", Interactive Multimedia Electronic Journal of Computer-Enhanced Learming, vol. 6(2), 2004
http://www.imej.wfu.edu/articles/2004/2/03/index.asp
6. Chi Chung Ko,, Ben M. Chen, Shaoyan Hu, Vikram Ramakrishnan, Chang Dong Cheng, Yuan Zhuang, and Jianping Chen, A web-based virtual laboratory on a frequency modulation experiment, IEEE Trans. On SMC – part c, 31(3), 2001
7. Virtual Microscopy laboratory at University of Delaware,
http://www.udel.edu/biology/ketcham/microscope/
8. L. D. Feisel, G. D. Peterson, "A Colloquy on Learning Objectives for Engineering Education Laboratories", Proceedings of the 2002 American Society for Engineering Education Annual Conference and Exposition, 2002.
9. J. T. Bell, and H. S. Fogler, "Virtual Reality Laboratory Accidents", Proceedings of the 2001 American Society for Engineering Education Annual Conference and Exposition, 2001.
10. SMSL-Smart Materials and Structures Laboratory at University of Houston,
http://rsmsl-1.me.uh.edu/
11. iLab at Massachusetts Institute of Technology, http://icampus.mit.edu/ilabs/
12. J. Harward, Jesus del Alamo et al., The iLab Shared Architecture: A Web Services Infrastructure to Build Communities of Internet Accessible Laboratories, Proceedings of the IEEE, Vol. 96, No. 6, pp. 931-950, June 2008
13. Java Programming Language, http://en.wikipedia.org/wiki/Java_(programming_language)
14. X. Chen, D. Olowokere and G. Thomas, "Teaching Java – Objects First with BlueJ", Proceedings of the 2008 American Society for Engineering Education Annual Conference and Exposition, 2008.
John Hopkins University, Signal, System and Control Java Applet, http://www.jhu.edu/~signals/
15. Adobe Flash, http://en.wikipedia.org/wiki/Adobe_Flash
16. National Instruments, http://www.ni.com
17. LabView Run-Time Engine 8.5, http://joule.ni.com/nidu/cds/view/p/id/861/lang/en