AC 2008-1316: REALISTIC LOOKING INTERFACES: IN SEARCH OF THE BEST ERGONOMIC METAPHORS FOR REMOTE AND VIRTUAL LABORATORY INTERFACES

David Olowokere, University of Alabama at Birmingham

Kayode P. Ayodele, Obafemi Awolowo University

Lawrence O. kehinde, Texas Southern University, Houston, Texas

Olutola Jonah, Obafemi Awolowo University

Temitope O. Ajayi, Obafemi Awolowo University, Ile-Ife, Nigeria

O.O. Akinwunmi, Obafemi Awolowo University, Ile-Ife, Nigeria

Realistic Looking Interfaces: in Search of the Best Ergonomic Metaphors for Remote and Virtual Laboratory Interfaces.

Abstract

In the last few years, remote and virtual laboratories have emerged as viable complements to traditional or legacy laboratories in fulfilling the experimentation component of engineering curricula. In fact, as we argue in this paper, remote and virtual laboratories have the potential to replace many legacy labs if the proper interface paradigms are identified. In this paper, we present the Realistic Looking Interface, which is a type of interface that focuses on the use of a metaphor that the student can more intuitively relate to a real-life laboratory. In essence, a student interacts with the laboratory using a point-and-click mechanism to manipulate graphical objects that have a one-to-one mapping to the real hardware backend. We present two implementation of a Realistic Looking Interface, one using a Java 3D space as a medium, and the other using 2D based flash vector representation. We show that in both cases, students psychologically relate the remote and virtual laboratories to real life ones and demonstrate a higher level of satisfaction with the experimentation process. We conclude that development and evolution of Realistic Looking Interfaces will allow virtual laboratories to eventually replace a large percentage of legacy laboratories.

I. Introduction:

Interest in Information and Communication Technology (ICT) -based alternatives to traditional laboratory experimentation systems has spiked in recent years. Of particular interest have been remote and virtual laboratories (RVL), which are systems where the Internet is used as a medium to allow users access remote laboratory hardware or software resources. RVLs are broadly classified under two categories: remote laboratories and virtual laboratories. Remote laboratories are systems whereby users are given access to real hardware backends located at a remote site. Virtual laboratories are RVL systems whose backend systems under test are implemented in software.

Various implementations of RVL have been described [1-9] and advantages ascribed to them have included ease of use, reduced safety concerns, scalability, availability, and minimal staffing requirement[9.] The various reported implementations have also demonstrated that RVLs can achieve, to a large extent, the thirteen objectives of experimentation in engineering education reported by Feisel and Peterson[10]. If RVLs have all those advantages over real labs and are able to impart the same experimentation skills to the same level in students, then one would expect that RVLs would be considered as valid replacements for real labs, rather than novel oddities that are deployed only in the absence of the real thing. In practice however, adoption of RVLs to date has been poor.

In this paper, we explore various issues relating to the ability of RVL to supplant real labs, and hypothesize that poor interface design is emerging as the most important limiting factor. We then

introduce a class of interfaces which we term Realistic Looking Interfaces and thereafter describe two implementations of the interface and briefly discuss students' response to them.

II. Constraints in Adoption of RVLs

One reason why RVLs have had problems replacing real labs is that there is often no *compelling* need for them to do so. Determining the number of experiments a student needs to acquire experimentation-related skill is not an exact science and so even when, for whatever reason, the recommended number of real labs is unattainable, the tendency is to simply make do with what is available rather than utilizing RVLs to supplement existing labs. If, for example, there isn't enough equipment available to carry out five Control Systems experiments, students would make do with the two or three experiments that are available. It is difficult to prove conclusively that a student who does only three laboratory experiments is less-trained than one who does five, and since most educators demonstrate high inertia to change anyway, there is no impetus to be adventurous in meeting the recommended experiment quota.

Another reason is that, perhaps because the field of RVL research is still relatively young, there is so much uncoordinated research and an almost total absence of universal standards. In fact, the problem is so much that there are no unambiguous definitions for even some of the most basic terms in the field. This leads to a situation where there is no dominant platform and as a result, the requisite large user-base that usually drives rapid development has not coalesced behind any specific platform. Most institutions that need to deploy RVL simply develop a new platform. Even in choice of implementation technology, there are myriad overlapping approaches and tools and the only tools that have attracted a noticeably large base of researchers in the field are LabVIEW and other National instrument software and hardware.

A third reason is that RVLs come with their own set of unique disadvantages and technical problems. For example, bandwidth limitations often hinder the effectiveness of RVL, since interface and usability considerations have to be constrained by bandwidth. In other situations, the prerequisite of having of computer-related skills sets the entry bar too high for students who are not sufficiently familiar with computers and software tools. Another crucial disadvantage is that many RVLs are fundamentally artificial. Either due to the fact the backend is simulated and thus less noisy and unpredictable than real systems, or due to approximations and errors introduced by instrumentation-communication infrastructure, RVLs tend to give results that could be at variance with real, local hardware. This problem of inaccuracy applies mostly to virtual laboratories but affects even remote laboratories unless specifically taken care of.

A fourth reason for the failure to date of RVLs to replace real labs is the perception that the experiences from the two categories of labs are different. There seems to be a generally held view that RVLs cannot deliver an experience as rich or as "real" as a real lab. For example, the general belief is that a student who performed his experiments using devices he never actually sees would somehow perform poorer than one who has actually seen and touched such equipments. The key to understanding this problem is of course to note that RVLs and the corresponding real labs engage a completely different set of senses. While real life labs usually engage the auditory, visual and kinesthetic senses equally, the RVLs have much lower kinesthetic and auditory components.

One curious fact, lost on many people, is the fact that RVLs may actually be more equipped to compete with real labs in today's evolving work environments which feature systems with frontend-backend architecture and software reconfigurability. For example, an electrical engineer emerging into the workplace today may never need to solder a discrete component throughout his career, but rather simply program reconfigurable devices like field programmable gate arrays (FPGAs). The truth is that the workplace itself is becoming less "real", and is depending more and more on the same types of techniques and technologies used in RVL. Even real labs are now using Virtual Instruments, which use the same software-mapped interface-to-functionality paradigm as RVLs. In other words, if the problems of RVLs can be sufficiently resolved, they could be even more appropriate than real labs for training engineers who will use technologies similar to RVL in their future workplace.

Most of the limiting factors to RVL adoption above have solutions in sight. Developing countries might actually be a vehicle for driving more rapid adoption of RVLs because in their case, there is a *compelling* need. Institutions in such countries usually operate with chronic underfunding, therefore they have started demonstrating a desire for technologies that could meet their students' experimentation needs within budget constraints. If third world institutions start adopting RVLs en masse, it is possible that by throwing their collective weight behind a few platforms, these would become de-facto standards due to the sheer user base. This should lead to better focused research and interoperability standards, solving the first two problems.

Many of the technical issues with RVL will inevitably resolve with time, and a large number of them can be solved by simply throwing more processing power and bandwidth at the problem, taking out some of the third category of problems mentioned above. In the case of inaccurate results, a solution already exists. Where accuracy and realism is very important, remote laboratories should be used rather than virtual laboratories. Unlike virtual laboratories that depend on software-actualized versions of a real system, a remote laboratory has a real system at the back end. Therefore, results from remote laboratories are not just as good as results from real systems, they *are* results from real systems that are simply not located in the same room as the user.

However, the fourth problem with RVLs, the ability of RVLs to impart the same kind of user experience as real labs, is a serious one that will not go away soon because researchers do not seem to realize how important it is. As the other limiting factors to RVL adoption disappear, this experience-impartation problem will become more and more central in determining how well students perceive an RVL. In a nutshell, users would have to be convinced that the experience of a student who uses an RVL for experimentation could be made to be just as "real" as one who uses a real lab, and that a student whose experimentation experience stems from purely RVL systems would acquire as much skill as one who used real hardware. We are convinced that this will not happen unless more attention is given to RVL interface design.

III. Interface Design in RVLs

A major reason why RVLs cannot yet offer an experience generally as rich as real labs is poor interface design in most RVLs. Countless studies have shown that the effectiveness of an

interface determines how useful a product is perceived to be[19, 20] and yet, in most RVL architectures, the interface is often designed as an afterthought. Researchers simply use whatever tools their choice of implementation technology constrains them to use, and the aim is often to come up with a minimalistic design that just works. In contrast, the lessons of interface design have for decades been incorporated into the bench top equipments that RVLs interfaces often try to copy.

Since most RVL interfaces are not designed by Human Computer Interface (HCI) experts, a lot of the basic design concepts in HCI are not addressed. For example, one of the best known interface design methods in HCI is user-centered design (UCD) [12, 19]. RVL interface designers do not usually apply UCD ideas even though doing so would ensure that an interface is much more effective. Figure 1 compares a typical RVL development process with a model that uses user-centered design.

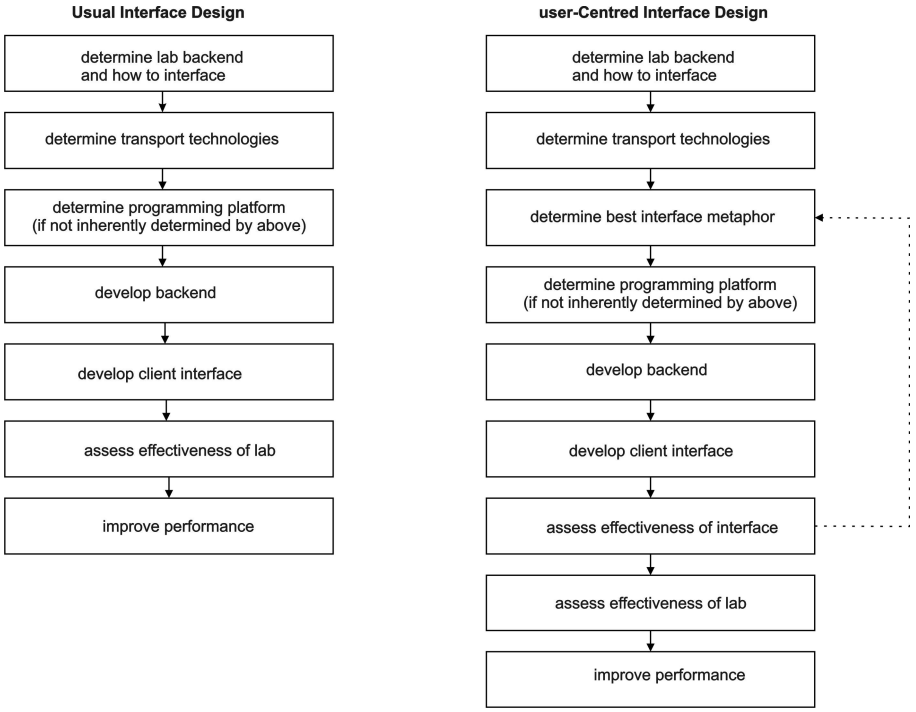


Figure 1: Generalized design steps for RVLs and proposed model incorporating user-centered design

One major difference between the two design approaches is that the interface metaphor is considered much earlier in the design process and in fact, determines what programming platform would be used where possible. In addition, after completion, the interface is evaluated separately from the RVL and if necessary, the metaphor would be changed. Obviously, the user-centered design would be more costly and time-consuming [13], but since lab development is a one-time process (albeit with periodic improvements) that once completed, gives rise to a lab that would typically be in use for years, the extra expenditure should be well worth it.

One significance of using a UCD approach to develop RVL interfaces is the fact that since the interface metaphor choice comes before the choice of programming platform, it allows an abstraction, separating the RVL technology itself from the interface choice. For example, an interface idea that works well in a similar context elsewhere can be penned down for

implementation for a new lab. Therefore, research on interface design can be done independent of lab implementation details. In this vein, a class of interfaces called Realistic Looking Interfaces (RLI) has been developed. In RLI interfaces, the emphasis is on using a metaphor that mimics the appearance and behavior of the real life backend system as much as possible.

One area where such interfaces would be very useful is in Electrical Engineering RVLs where students need to interact with backend circuits. Many current RVL implementations use a metaphor that presents the RVL to the student using a schematic representing the device or system under test [1, 2,3,4,9]. The problem is that the student has seen real electronics components, has seen circuits drawn in books or simulation packages and has been taught to see a schematic as just shorthand representation of the real thing. In a real laboratory, students interact not with schematics but with solid components but interestingly, most Electrical Engineering RVL designers choose to employ interfaces that use schematics rather than more direct representations of real components. When the student sees a schematic metaphor, he subconsciously sees the entire lab as not being "real", but a "representation of the real thing". In contrast, RLI aim to make the student see the lab interface as a local representation of real devices that are at a remote location.

IV. Realistic Looking Interface Example: Flex OpLab RLI

To test the effectiveness of the RLI concept, an operational amplifier lab ('Flex OpLab') was developed with a 2D vector Flash interface. A block diagram of Flex OpLab is presented in Figure 2. The lab uses a modified version of the Massachusetts Institute of Technology (MIT) iLab architecture [2, 3, 4]. Whereas the iLab architecture normally has three tiers (the Client, the Service broker and the Server tiers), the Flex OpLab omits the Service Broker, rather allowing direct Client-to-Server communications using web services.

The Server component of the Flex OpLab is made up of a web service, a database, an experiment execution engine and the hardware at the backend. By consuming web service methods implemented by the server, the Client can carry out activities like submitting an assignment and

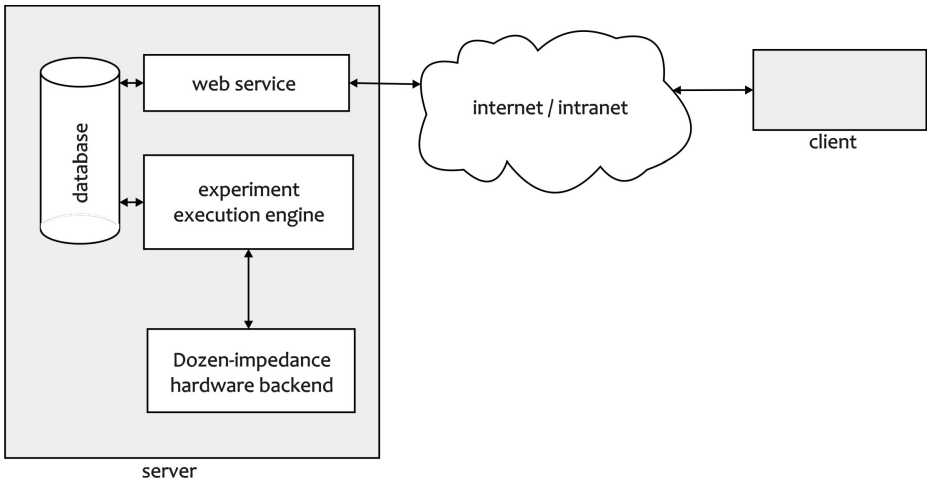


Figure 2: The Flex OpLab

retrieving corresponding results. If for example, the *Submit()* method is invoked, the web service interface receives the experiment specifications from the Client and copies them into the database. It also sets a flag indicating that a new experiment has been submitted. The experiment execution engine continually monitors the database queue and once the new submission is noted, it communicates with the hardware, setting it up as appropriate, and reading the relevant output signals.

The hardware backend of the Flex OpLab is a Dozen Impedance operational amplifier configuration[18] which allows a small circuit made up of an operational amplifier, a dozen impedances, and some switches, to be configured as any of a small set of circuits (Figure 3). For example, by closing switches s_2, s_3, s_{12} and s_{15}, and substituting appropriate capacitors for impedances Z_2 and Z_3, the circuit becomes an Integrator or a Differentiator. An LM 324 operational amplifier was used along with a custom-built switch matrix with MAX4664 solid state switch arrays that have a maximum ON resistance of 5 Ohms.

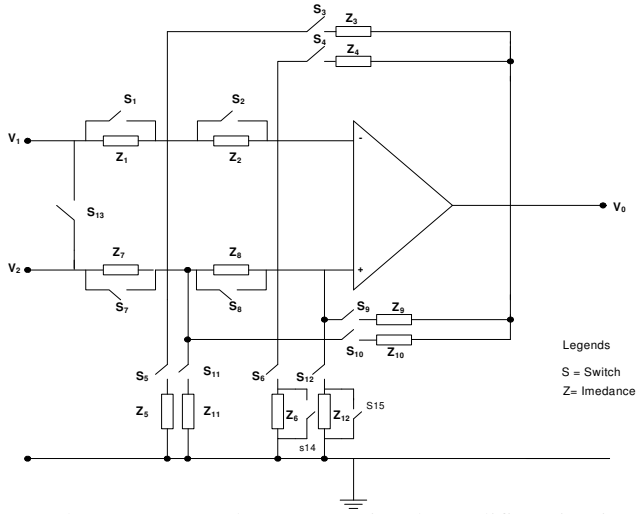
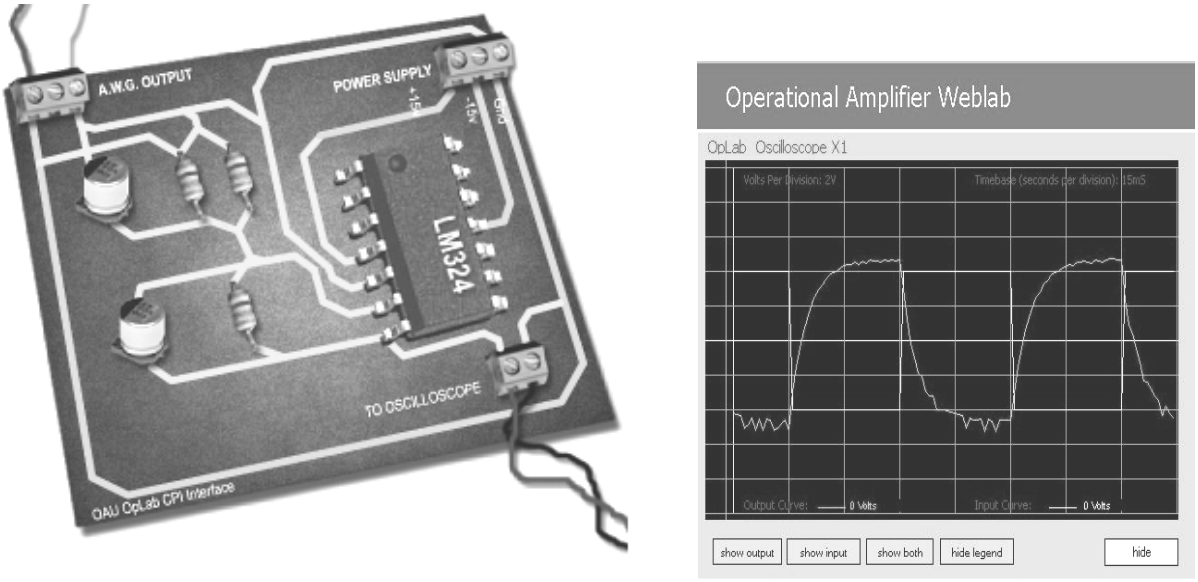


Figure 3: The Dozen Impedance Operational Amplifier Circuit

The Client is implemented as a Flash control using the Adobe Flex platform. Flex is a new programming platform that allows the creation of programmatically-driven platform-independent Flash files that can be embedded in web pages. One of the reasons why this medium was chosen is because the Flash plug-in is already available on most computer systems on the Internet[14] and so would not require any additional software installations by the student to use. Illustrating the core ideas of RLI, the Flex Client represents the backend circuit as a printed circuit board (PCB) on which the various components of the circuit are laid out (Figure 4a).

To access the Client interface, a user would log in and select the OpLab from a menu. Thereafter, a web page is presented, with the Flash Client embedded. For a typical experiment, the student would be required to configure the circuit as a simple operational amplifier circuit like an inverting amplifier or an integrator. The student would need to determine the appropriate nodes to connect. To connect any two nodes on the circuit, students would click on the right pair of terminals. A green PCB trace appears between the two terminals indicating that a connection has been made.



(a) (b)

Figure 4: (a) Main interface and (b) signal plotting interface of the Flex OpLab RLI

Connecting the nodes of the circuit allowed another advantage of Flash to emerge. Flash allows the seamless integration of different types of media from video to images and simple text. In the present instance, the use of a RLI implies that the students need to determine what physical pins on the LM324 correspond to what functionality. To achieve this, the students are given access to a datasheet which is implemented as part of the Flash file.

Once the student has finished configuring the circuit, he hits the "Submit" button. The Client thereafter generated an experiment specification file from the student's selections and invokes the *Submit()* method of the Server web service to submit the experiment for execution. After execution is complete, the *Notify()* method of the Server web service allows the Client to plot the output on the screen (Figure 4b)

V. Realistic Looking Interface Example: Materials Strength Investigation Lab

a. Video Games as Motivators For RVL Interfaces

All other things being equal, an immersive interface would be more engaging that a non-immersive one[15] and immersion is one of the hallmarks of video games. Investigating how video games can be used for educational purposes is a research problem that has received considerable attention recently[11, 16, 17] .The motivation for this attention is the fact that the present generation of students are completely different from older generations, and most of what they know about the world and how they interact with it are bound with computers, the Internet, and multimedia. For example, it has been estimated that by the time a typical student nowadays is 21 years old, he would have spent 10000 hours on video games, 10000 hours on the cell phone, 20000 hours watching the television and would have sent 200,000 emails and Instant Messages, while spending less than 5000 hours of book reading[11]. Manufacturers are constantly redesigning

product interfaces to take cognizance of these changes in the aptitudes and skill sets of this generation and it is vital that educators in general and RVL developers in particular also make appropriate changes.

Of particular relevance to RVLs interface design is the category of games known as Massively Multiplayer Online Games (MMOG). These games offer the best semi-immersive interfaces currently on the Internet and generally allow users to interact but their objectives and point earning rules make them unsuitable for use as RVL. However, in recent years, a new category of MMOG has emerged. In these new MMOG, there is less focus on the accumulation of points from game play, but rather, players are allowed to define their reasons and rewards for playing. The most popular game in this category is Second life (SL). It is a virtual world where users can interact with hundreds of thousands of others. One of the most important characteristics that differentiate SL from other MMOG is that the world is almost entirely driven by content developed by residents. Thus, SL could potentially combine the social interaction offered by MMOG with the ability to develop interfaces that connect to real devices or simulation engines. This potential is not entirely achievable right now but the framework is already in place.

In SL, residents interact with others through their avatars and usually observe most of the societal protocols of interaction. The one prohibitive limitation of SL as an RVL interface is that connecting objects in SL to real backend objects in the real world is presently impossible. The SL environment allows scripting, which could enable development of basic simulation-only laboratories but even then, the capabilities are limited.

To study what aspects of SL would be valuable to an RVL interface, a group of students were asked to meet in SL and carry out specific group-oriented tasks. They were asked to arrange set of boxes in pairs, and each pair was timed. By studying how the students carried out the exercise, we gained a few insights to develop a 3D semi-immersive interface for a virtual laboratory which we called Materials Strength Investigation laboratory.

b. Overview of Materials Strength Investigation lab RLI

The Materials Strength Investigation (MSI) lab interface is a 3 dimensional (3D) virtual world created using the Java 3D API, and developed to mimic Second Life as much as possible. We mimicked the SL look because what we would have liked to do was to develop the interface within SL. We only resorted to a Java 3D world because we could not couple SL objects to real backends. We thus develop our Java interface to combine the SL environment we like and the ability to interface to real life devices. The MSI Java 3D virtual world was built to serve as the client tier in a multi-tiered remote laboratory architecture. This client tier was to provide a realistic laboratory experience. The quest for realism gave rise to these major requirements:

- Realistic Appearance
- Interactivity
- Animation
- Communication over a Network

Interactivity gave the users the ability to alter some of the visible elements of the MSI world in various ways. The users interacted with the elements by using the keyboard and Mouse.

Animation gave the elements the ability to change their appearance and behavior in response to stimulus. Communications over a network allowed the interface send and receive data to and from the lab server.

In building a 'realistic' MSI-world using the Java 3D API, a virtual universe was created. The components of this virtual universe (MSI-world elements) were 3D Java objects which loaded their geometry from models. These geometric models were created using non-Java tools (e.g. Wings 3D), written to files and loaded dynamically. The geometric models were provided in two flavors:

- As local (to the client machine) data files, and
- As data sent over the network from the server.

The geometric models were dynamically (at runtime) loaded into Java objects, by using the appropriate *loader* object for each model format (eg. *ObjectFile* object for the *.obj* model format).

In meeting the requirements for interactivity, the *Behaviors* objects were utilized. These objects (some from the Java 3D API, and some custom made), alter the *nodes* of the virtual universe's *scene graph* in response to a variety of stimuli. The stimuli for behaviors were primarily keyboard and mouse events as well as changes to data. Alterations in position, appearance and other attributes and properties of elements, in response to stimulus provided a well animated virtual world (MSI-world). When necessary, special *Interpolation-Objects* were added to the scene graph, to provide for better animation.

Communications with the lab server was hidden behind an abstraction layer. This abstraction layer was implemented as a *singleton* class object. This class utilized objects from the Java network programming packages, and implemented a custom designed interface called *IMSIWorldServer*. This interface allowed for loose coupling between experiment engine implementations and the MSI-world.

The general structure of the MSI lab interface is illustrated in Figure 5.

VI. Students' Response

While the Flex OpLab has been completed (partly because the hardware had been previously developed for another project), the MSI lab is still under development. Despite this however, it has been possible to assess students' response and it has generally been positive.

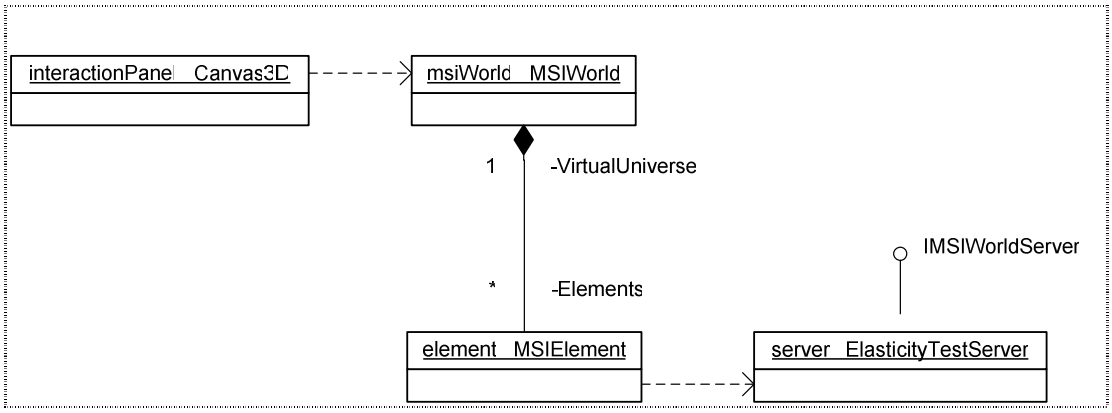


Figure 5: MSI Lab interface structure

As a simple test of the effectiveness of the Flex OpLab RLI, eleven students were grouped into two sets. One set was made to carry out the lab using the RLI interface, while the other set used the classical OpLab interface (Figure 6). Then, the groups were switched and each now worked with the other interface. Each student was then asked to use a simple ratio to describe their preferences. Every single student rated the Flex interface higher than the classical OpLab one. In fact, only one student rated the interfaces less than (65:35) in favor of the Flex RLI.

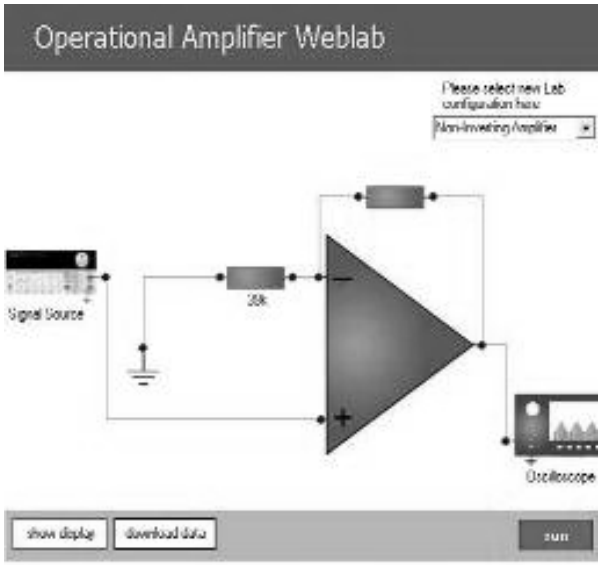


Figure 6: Classical interface of the OpLab laboratory

After using the Flex RLI, the students were asked to try out the Materials Strength Investigation lab. After using it, they were asked to compare the two interfaces as they did with both OpLab interfaces. Somewhat surprisingly, just over half of the students (7) gave the Flex OpLab interface a better score than the MSI lab. All the students who rated the Flex interface higher however admitted that the incomplete state of the MSI interface influenced their rating. The exercise will definitely be repeated once the MSI interface is completed. It is however instructive that every one of the 11 student rated both the Flex OpLab interface and the MSI Java 3D interface as being better than the schematic-metaphor classical OpLab interface.

VII. Conclusion

Despite the recent interests in RVL, some factors continue to limit the rate of their adoption. While a lot of these factors are being resolved, effective interface metaphors will probably remain the most important single reason RVLs will not see increasing use in place of real labs. By applying the rules of HCI, better interfaces can be developed. We have shown that using the Realistic Looking Interfaces for example, the perception of students about RVLs can be improved. Once students' perceptions are improved, they will be more confident in the ability of RVLs to effectively meet the objectives of laboratory experimentation.

VIII. Acknowledgement

The authors would like to thank Jesus del Alamo, Steve Lehman, Jud Harward and members of the Center for Educational Computing Initiatives (CECI) at Massachusetts Institute of Technology for assisting with various aspects of the research. Some of the components used for this research were donated by Maxim Semiconductors. The Java 3D interface and other programming components were developed by Dayo Omitayo. The research was funded by a grant from the Carnegie Corporation of New York.

Bibliography

1. Saliah-Hassane, H., Dumont-Burnett, P. and Kedowide, C. (2002): A Framework for A Distributed-Measurement User Interface. Proceedings of the International Conference on Engineering Education, Manchester, UK.
2. Del Alamo, J.A., Chang, V., Hardison, J., Zych, D., and Hui, L.(2003): An Online Microelectronics Device Characterization Laboratory with a Circuit-like User Interface, Proceedings of the International Conference on Engineering Education, Valencia, Spain.
3. Del Alamo, J.A., Hardison, J., Mishuris, G., Brooks, L., McLean, C., Chan, V., and Hui, L. (2002): Educational Experiments with an Online Microelectronics Characterization Laboratory, Proceedings of the International Conference on Engineering Education, Manchester.
4. Viedma, G., Dancy, I.J., and Lundberg, K.H.(2005): A Web-Based Linear-Systems ILab , Proceedings Of The American Control Conference,. Vol. 7, Issue 8-10, pp 5139 – 5144.
5. Saad,M., Saliah-Hassane, H., Hassan, H., El-Guetioui, Z., and Cheriet, M. (2001): A Synchronous Remote Accessing Control Laboratory On The Internet. Proceedings of the International Conference on Engineering Education, Oslo, Norway.
6. Berqia, A., Diop, A., and Harms, J. (2002): A Virtual Laboratory for Practical exercises. Proceedings of the International Conference on Engineering Education, Manchester, UK.
7. Godfrey, J., Zhang, J., Ball, A., and Adams, R.(2007): Implementing A Remote-Access Engineering And Technology Laboratory Through A Graduate-Level Team Project. Proceedings of the 2007 American Society for Engineering Education Annual Conference & Exposition
8. Bischoff, A., and Rohrig, C.(2001): Remote Experimentation In A Collaborative Virtual Environment. Proceedings of the 20th World Conference on Open Learning, Dusseldorf, Germany.
9. Del Alamo, J. A., L. Brooks, C. Mclean, J. Hardison, G. Mishuris, V. Chang, And L. Hui, "The MIT Microelectronics Weblab: A Web-Enabled Remote Laboratory For Microelectronics Device Characterization." 2002 World Congress On Networked Learning In A Global Environment, Berlin (Germany), May 2002.
10. Feisel, L., and Peterson, G.D. (2002): A Colloquy on Learning Objectives for Engineering Educational Laboratories, Proceedings of the 2002 ASEE Annual Conference and Exposition, Montreal, Ontario.
11. Oblinger, D. (2004). The Next Generation of Educational Engagement. Journal of Interactive *Media in Education*, 2004 (8). Special Issue on the Educational Semantic Web. Issn:1365-893x

12. Vredenburg, K., Isensee, S., and Righi, C. (2001). User-Centered Design: An Integrated Approach: Prentice Hall.
13. Beale, R. and Bordbar, B. (2005) "Using modeling to put HCI design patterns to work", in HCI International. 11th International Conference on Human-Computer Interaction, Lawrence Erlbaum Associates, Inc (LEA). Las Vegas, Nevada, USA
14. Emigh, J.(2006) "New Flash Player Rises in the Web-Video Market," *IEEE Computer*, vol. 39, no. 2, pp. 14-16, Feb., 2006
15. Leigh, J., DeFanti, T., Johnson, A., Brown, M., Sandin, D., (1997) "Global Tele-Immersion: Better than Being There", In the proceedings of 7th International Conference on Artificial Reality and Tele-Existence. Tokyo, Japan, Dec 3-5, 1997, Pp 10-17.
16. Gee, J. P. *What Video Games Have to Teach Us About Learning and Literacy*. New York: Palgrave/Macmillan, 2003.
17. Squire, K. (2003). Video games in education. International Journal of Intelligent Games & Simulation, Last retrieved January 5, 2008: http://www.scit.wlv.ac.uk/~cm1822/ijkurt.pdf
18. Kehinde, L.O. (1989): *The "Dozen-Impedance" Operational Amplifier Module for Experimentation*. International Journal of Electrical Engineering Education, vol. 26, No. 3, 1989, pp 224-232, Manchester UP.
19. Bevan, N. (1995): Usability is Quality of Use. Proceedings of the 6th International Conference on Human Computer Interaction, Yokohama, July 1995. Anzai & Ogawa (eds), Elsevier
20. Hilbert, D. M. and Redmiles, D.F. (2000) ACM Computing Surveys, Vol. 32, No. 4, December 2000, pp. 384–421.