Remote Labs Accessible through 3D environments

A Case Study with Open Wonderland

David Costa[1], Gustavo Alves[2], Paulo Ferreira[3] and Juarez Silva[3]
[1] ISEP/Informatics, Porto, Portugal
[2] ISEP/Electrotechnics, CIETI-LABORIS, Porto, Portugal
[3]ISEP/Informatics, CIETI-LABORIS, Porto, Portugal
[4]UFSC/Informatics, Araranguá, SC, Brazil

***Abstract*— 3D immersive environments have been spreading through domains such as enterprise activities, education and entertaining, due to the Second Life growth. This and other 3D environments can be used to increase the sense of immersion of users accessing remote laboratories, namely when controlling & observing measuring instruments. This paper contributes to the overall research efforts, on this domain, by presenting a case study of how to remotely access a digital multimeter, through a 3D environment developed in Open Wonderland.**

***Index Terms*— Remote Labs, 3D environments, measuring devices.**

I. INTRODUCTION

A remote laboratory is a type of lab, which contains measuring and/or other electronic instruments that can be remotely accessed or controlled, using web pages or other applications. One of these applications can be immersive environments also known as 3D environments. These provide online 3D virtual worlds where users can navigate, add content, or communicate among them. A 3D environment usually has a server that provides virtual worlds to users, who access it using a web browser or a specific application downloaded from the Internet. It adds that combining 3D environments and remote labs can positively impact the sense of immersion of users. Although a number of example cases have already been reported in literature, not all combinations have been tried, whereas the features of a particular 3D environment may turn the developers' task of providing remote access to a Weblab or a particular measuring equipment more or less difficult, if not impossible at all. We contribute to this on-going effort by presenting a case study based on using Open Wonderland, a 3D environment developed in Java for remotely accessing a digital multimeter.

The rest of the paper is structured as follows: section 2 provides a general overview on 3D environments and Weblabs, namely the main features of a 3D environment and a number of examples of Weblabs accessible through 3D environments, as reported in literature. Section 3 presents the case study implementation. Section 4 presents the test scenarios and the results obtained. Finally, section 5 concludes the paper and points out some future work.

II. 3D ENVIRONMENTS AND WEBLABS

For better understanding the existing examples of Weblabs accessible through 3D environments we first present some features of this sort of web-based applications.

A. Features

A 3D or immersive environment is an online simulated virtual world available to Internet users. Its concept is based in the next features [1]:

- **3D interface:** its appearance should be as similar as possible to the real world.
- **Avatar based:** it is the user's 3D environment representation. A user navigates in a virtual world with his avatar. It is a dummy apparently similar to a person. Many 3D environments provide configuration features to the users' avatar such as: sex, clothes, hair or skin colour, etc. The avatar makes a user to be recognizable by others in a virtual world.
- **Immediate communication:** users can communicate among them using chat features. Some 3D environments provide text, voice or video communication.
- **User content creation:** a 3D environment should provide user content creation features. This content is usually based in 3D objects, to be added to the virtual world. Most of the times, these objects are real world common ones reproductions. In some 3D environments this user created content is used for trade, as some environments have their own inside economy.
- **Web access:** a 3D environment virtual world should be accessible to users from a web browser or a client application available on the Internet.
- **Persistence:** a 3D environment virtual world should continue to exist whether or not users are logged in.

B. Weblabs accessible through 3D environments

The first Weblab example is the SecondLab [2], developed at the University of Deusto, Spain. It is a Weblab which allows the remote control of microbots from a Virtual Laboratory (VL), developed using the Second Life 3D environment. The SecondLab uses the Weblab-Deusto, a remote lab that provides experiments to its users, who use it as a tool. This remote lab has a user queue according to the experiment. In these queues, a time period (e.g. 200 seconds) is reserved to each user to do the

experiment. One of the experiments available is the microbots remote programming, accessed by SecondLab. The SecondLab establishes the VL interface to the Weblab-Deusto in order to remotely control the microbots. This Weblab concept seems to be flexible in order to add more remote experiments, if provided by the Weblab-Deusto. The SecondLab working is highly dependent on the Weblab-Deusto performance and working, if it fails it will cause a fail on the SecondLab. Adding features to this Weblab implies costs, as the Second Life programming tools are not free.

The RexLab V Virtual World [3] provides a 3D interface developed in the Open Simulator (OpenSim) 3D environment. It has a VL or classroom allowing interaction among students and teachers. Its goal is to provide devices' remote control in physics subject experiments. It interacts with Moodle and Sloodle platforms, where Moodle is used to store and manage educational content, and Sloodle establishes a bridge between Moodle and OpenSim. This detail is very important to organize the educational content and aids the teaching preparing task. This Weblab may require a more careful user management in order to distinguish the users, not only by its id or interaction.

The TEAL Studio project [4] is also related to physics subject experiments. It is a Weblab which provides a 3D collaborative virtual learning environment, where users can remotely do the experiment "Force on a Dipole", which is a remotely available magnet that allows students to configure some electric variables in order to collect the experiment data. The experiment is accessed by the TEALSim application that is a simulation toolkit. This Weblab was developed using the Project Wonderland 3D environment, which is the user 3D interface which makes the VL where users can observe the "Force on a Dipole" experiment. The VL implementation uses the iLab Project, which provides a tool for designing Internet accessible lab experiments. Both the TEALSim and the iLab were developed at the MIT. The Project Wonderland and the TEALSim were developed in Java; this made the programming task easier and costless, as the Java and Project Wonderland API are free. Java developing applications may require powerful hardware in case of providing many 3D interface features.

The Virtual Enterprise Lab 2.0 [5] is a virtual university campus with several VL, classrooms and halls places, which does not contain remote experiments. It was developed using the Open Wonderland[1] 3D environment at the Lucerne University of Applied Sciences and Arts, in Switzerland. Its developing is a second version, which derivates from a first one developed by students in their bachelor diploma thesis. This WebLab is mentioned here because the whole virtual space allows the addition of 3D objects or virtual devices capable of remotely controlling real ones, due to some Open Wonderland 3D environment features. This example offers a complete virtual learning space capable of implementing remote experiments. It may require powerful hardware in case of providing many and/or complex experiments.

The Vetunimi [6] Weblab is a veterinary medicine VL, developed in Italy and built using the Active Words 3D environment, one of the oldest 3D environments. It is used for veterinary medicine teaching. The Vetunimi provides a VL place similar to a veterinary clinic, where users can remotely control devices used for biomedical analysis. This Weblab points out the experiments required for its subject in a clinic context, unlike others that focus essentially on a single example in one subject, to purpose a single methodology. Its goal is knowledge sharing by making tutorials and simulations available to complement the learning process. Currently, this Weblab VL appearance details are below the most recent Weblabs, as this one was developed in 2001, before the existence of the most common 3D environments.

All this Weblabs' research is the first step to decide which 3D environment will be used in a case study implementation. To do this task it is required a 3D environment easy to program in order to save developing time in building additional contents. Its programming tools should be free to imply no costs. The most adequate 3D environment which complies with those requirements is the Open Wonderland (OW). Its developing programming language (Java) makes this 3D environment one of the easiest contents adding and programming. The case study developing is costless as it will only use free API from Java and OW. There are many free Java code libraries available on the Internet, which aids the developing of features that would not be possible just with the use of the Java and the OW API. The OW has a well organized, easy access and easy reading documentation, which is very helpful in installing, configuring and managing the 3D environment and its features/contents. This detail is important for an easier learning process of people who work with the OW for the first time.

III. A CASE STUDY IMPLEMENTATION

In order to validate the potential use of OW for granting access to remote labs, it was implemented a simple case study where a remote user is able to control/observe a real multimeter. The case study implies, in a first stage, developing a physical and logical architecture, which allows a remote user to access the multimeter. This architecture is broke down on its individual components, which includes a real multimeter virtual representation, later characterized in terms of its features and access mechanisms.

A. Architecture

This case study is supported by a network infrastructure, formed by several components, which represent the Physical Architecture of our case study. It depends on the devices software applications. These ones and its interactions through the different devices, form the case study Logical Architecture. Both the physical and the logical architectures are shown in Fig. 1.

[1] Open Wonderland is a more recent version of Project Wonderland

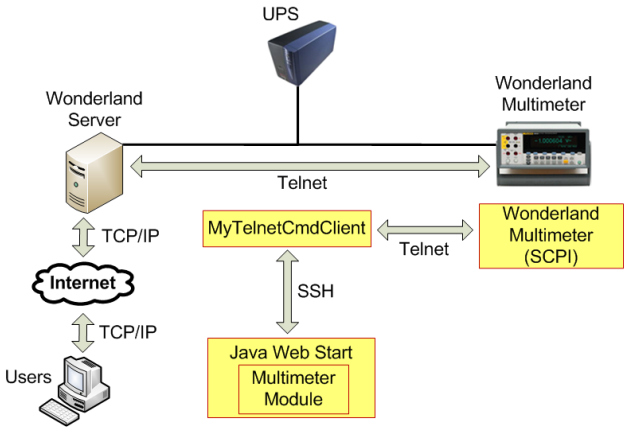

Figure 1. Case study Physical and Logical Architecture

Within the Physical Architecture each component has its functions, namely:

- **Wonderland Server (WS):** is the server where the VL runs. It stores the OW server with a virtual world. This one has laboratory features. The WS is a PC with a 1.6 GHz AMD Sempron CPU, 1 GB RAM and Ubuntu 9.10 Operating System (OS). It complies with the OW server minimum requirements [7]. The OW server runs on top of that OS.
- **Wonderland Mutimeter (WM):** is the real multimeter (RM), a Fluke 8845A.
- **Uninterruptable Power Supply (UPS):** prevents the WS abrupt shutdown, during a limited period of time, long enough for a clean shutdown sequence.

Users control the RM device, by accessing the WS through an Internet connection (TCP/IP). The WS has an application capable of reading data from the RM, through a telnet connection. When that application runs, the WS gets data and retrieves it to the user, who controls the virtual device in the OW VL.

Each component in the Physical Architecture runs a specific application which provides reading data from the RM to users. The user's computer runs the VL, where the virtual device is. When the user presses one of its buttons, it runs an application named *MyTelnetCmdClient*. This one is in the WS and it is OS command line based. It reads data from the RM, using a telnet connection, and is executed by a remote command. This application is triggered by the virtual device, using a SSH connection between the user's computer and the WS, which runs a SSH server beyond the OW one. These operations are supported by the Logical Architecture of our case study example.

In that architecture is shown the data value read operation path from the RM to the virtual device in the VL. This device or virtual multimeter (VM) is represented by the Multimeter Module. It is located in the virtual world, provided by the OW client application Java Web Start (JWS), which runs in the user's computer. The VM establishes a SSH connection to the WS, which runs the *MyTelnetCmdClient* application. This one executes a SCPI command in the RM. The value read is then retrieved to the VM.

This SSH connection allows to avoid the RM device inherent availability to the Internet, due to the telnet protocol weak security. That device is only accessible directly by the WS in its network.

B. *Structure and Components of the 3D environment: Open Wonderland*

A user accesses to an OW virtual world, by accessing the host where the OW server runs. The OW 3D environment consists in a server, which provides virtual worlds. This server is installable in any PC, similar to the one used in our case study. An OW server requires the Java Virtual Machine, Apache Ant and the Subversion software installed in the PC's OS. The installation is based in two essential steps [8]: the first one is downloading a 3D environment source code copy, from the OW online repository, using the Subversion software. The second step is compiling the source code with the Apache Ant. When compiled, the OW server is ready to start in the OS. It is also started with the Apache Ant. After the server being compiled, configured and started, it can be accessed through a Web page. This page will show the OW server initial one [9], where users can download the JWS client application and administrators can access the server administration page. The JWS application allows users to navigate in the virtual world; it requires a client PC with the Java Virtual Machine installed and a video card supporting OpenGL with at least 128 MB video memory, beyond the minimum requirements to run an OW server.

All OW server administration and management is done in Web pages, by System Administrator users. The main OW server features to administrate are: server components, users, virtual worlds and modules. An OW server has four components [10]:

- **Web Adminstration Server:** supports all the server administration/management.
- **Darkstar Server:** provides the virtual worlds to users.
- **Shared Application Server:** provides applications running in the OW server's OS to be used inside a virtual world.
- **Voice Bridge:** provides voice chat features between users.

An OW virtual world in this case study is only available to users stored in OW server. It was added a specific user, just for testing the VL and the VM features. It is available using the next data:

- Address: http://www.laboris.isep.ipp.pt:8080
- Username: usertest
- Password: wonderland123

The users' feedback is always important to the success of every application.

C. *Virtual Laboratory Features*

The VL in Fig. 2 is one of the virtual worlds provided by the OW server. Only an OW System Administrator user can change the virtual world which is going to be available to users. These access the VL set in OW server, using the JWS client application. This VL (Fig. 2) is a 3D scenario with trees on the horizon, grass on the ground,

and a table with the VM on top of it. The VL has also a PDF file, which is the RM user's manual. Its purpose it is to show the real device's features and how it can be used, in order to users know what device they are remotely controlling from the VL. Both the horizon trees and the ground grass are KMZ models files downloaded from Google 3D Warehouse. They were put in the virtual world, by a simple file drag & drop to the JWS window. The virtual device, the table and the PDF file are OW modules, added to the virtual world. The OW 3D environment is made of modules, which are software components. These are the additional content basis, in order to add or enhance features to the OW or to its virtual worlds. Positioning and dimensioning its elements configure the virtual world aspect depicted in Fig. 2. The dummy present in that figure is the user's *david* avatar, i.e. the author's avatar.

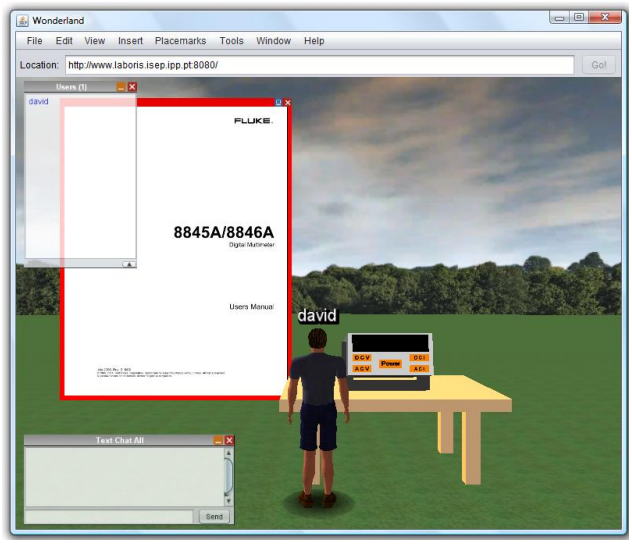

Figure 2. VL World

D. *The Measuring Device*

In the VL, there is a virtual measuring device, which is the VM. This one is a multimeter, a device which measures electric values, when linked to an electric circuit, such as voltage and current. The Fluke 8845A model is remotely accessible by an Ethernet interface. Its reading values are remotely got by SCPI commands, through a telnet connection. If a user specifies the RM IP address and its port, he/she is able to write SCPI commands to get measured values. The RM device just supports one remote connection at a time. A multimeter is one of the most common and easy using equipment in an electronics laboratory.

The virtual device in our case study only provides the variables voltage and current both in AC or DC measuring modes, in a first stage. Its graphical aspect was programmed using 3D objects (boxes and labels) from the JME code library. The remote connections features are based in code from the Ganymed SSH-2 [11] library, which has functions to run a command in the WS, through a SSH connection. This command retrieves the measured value. It is provided by the *MyTelnetCmdClient* application, which establishes the telnet connection to the RM and executes the SCPI Command for the measuring variable requested. This application uses the Apache Commons Net [12] code library, to implement telnet connection features. The VM and the *MyTelnetCmdClient* application were developed using the Java programming language. The measured value should be the same and appear in both virtual and real devices display, as it is shown in Fig. 3. The VM has five buttons, each one with its features:

- **Power:** turns the device on/off. When first turned on it measures the voltage in DC mode, as it is done by default when the RM is turned on.
- **DCV:** measures voltage in DC mode.
- **ACV:** measures voltage in AC mode.
- **DCI:** measures current in DC mode.
- **ACI:** measures current in AC mode.

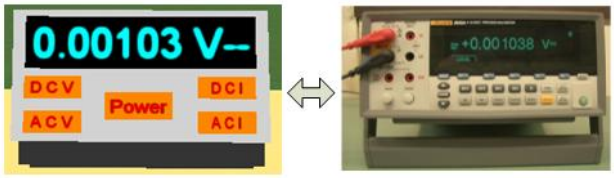

Figure 3. Virtual and Real Representations of the Measuring Device

When a button is released, it executes a command with the appropriate argument according to the measured value. The command is the *MyTelnetCmdClient* application, executed in the WS, through a SSH connection.

As the VM mimics the RM, the DCV, ACV, DCI and ACI buttons only work when it is on. This virtual device is capable of storing its state (on/off) and the last measured value, if the last user leaves the VL, with the VM on.

E. *Access Mechanisms*

Like a real device, all VM buttons only work when the user's avatar is close enough to it. The distance is calculated using a linear distance formula, based in 3D orthogonal coordinates (XYZ). The VM is accessible to all users stored in OW server. Each one watches its measures and other users' ones. It has a concurrent access mechanism to avoid conflicts when several users are intending to use it. This mechanism consists in locking the VM exclusively to the first user, who turned it on. He/She has the VM control and the others are not able to use the device, even if they push its buttons at a close distance. If other users intend to control the VM, they should ask for it, using the OW chat features. When a user asks for the VM control, the user who has its control should decide whether or not to transfer it. If he/she decides to do so, he/she should turn the VM off and the control is released. Then the user who asked for it, should turn the device on, and take over the control.

The concurrent access mechanism is prepared to transfer the VM control to the user who first pushes a button if the user who had it before leaves the VL with the device turned on. This feature prevents a VM dead lock. If not implemented, in case a user left the VL, while controlling the VM and did not turn it off, then it would not be possible to transfer its control. That would mean

other users could not use the VM, unless the controlling user returned to release it.

IV. Test scenarios and Results

All features described in the previous section were submitted to several conditions present in a case study usual using. When those conditions are grouped in a predefined set, it establishes a test scenario. In Table I, there is a features set to be tested, in each test scenario. These are described in the next subsections. Common situations are how the JWS works according to its computer specifications and network location and how many users are using the VM, as this device was developed to run in multi-user scenarios. Planning a test scenario is considering these situations in order to test certain 3D environment or VM features. For each test scenario it is defined a success criterion that defines the tests results, in order to check the case study usability.

TABLE I. CASE STUDY TEST SCOPE

Test scenario	Testing Feature	Success criteria
Computer where the JWS application runs (Specifications and OS)	Virtual world 3D objects detail level	Low
	Virtual world loading time	≤ 30 seconds
	Virtual world using response delay (includes VM using)	≤ 3 seconds
	VM features	Get a measured value, by pressing all VM buttons
Number of users	VM accessing mechanisms	VM control transfer through users
		VM lock to users, who do not get the device control
Network node, from where to access the OW server (Distance)	Virtual world loading time	≤ 35 seconds
	Virtual world using response delay	≤5 seconds

A. *Using the 3D environment*

This test scenario goal is to check the 3D environment usability, by the JWS application behavior in a set of four computers. These have different hardware specifications, OS and will access the VL from the case study local network or from another one.

In this scenario, just one user accesses the VL world (Fig.2) at a time. He/She has to use all the 3D objects (VM and PDF document), which include user interaction. The test will be successful if the virtual world loads in 30 seconds or less and the 3D objects response delay is 3 seconds or less. These times were defined considering an acceptable maximum duration that a user can wait. Any duration longer than those makes the 3D environment using inappropriate.

This test was successful in all computers due to the virtual world 3D objects low detail level, without compromising its appearance. A low detail level does not require powerful graphic hardware, as the virtual world was built to be available to as many users as possible.

Table II lists the computers used in this scenario, indicating its hardware main specifications and OS. All comply with the minimum requirements to run the JWS application, except computers 2 and 3, which have a 64 MB graphic card memory below the 128 MB minimum required to run the JWS. In computers 2 and 3 the JWS status messages window becomes similar to Fig. 4 when activated, due to insufficient video memory. In that window, the warning message "Forcing Low Detail" appears, which indicates the JWS is forcing the 3D objects detail level to be as lower as possible in order to make them available. Fig. 4 shows the JWS status messages window activated in computer 2. A high amount of messages is indicated in the figure circled scroll bar size, due to scarce hardware resources.

In computers 2 and 3 the virtual world loading time is longer than in computers 1 and 4. Computer 4 was used to check the JWS working in Linux systems. In this case the avatar motion is slower than its correspondent, in other computers, but does not compromise its utilization.

TABLE II. TESTING COMPUTERS SPECIFICATIONS

Computer Model	CPU Model/Clock rate(GHz)	RAM (GB)	Graphics card model (Video memory)	OS/ Version
Toshiba Satellite A300-1ID (Computer 1)	Intel Core 2 Duo 2.5	4	ATI Mobility Radeon HD 3650 (512 MB)	Windows Vista Business SP2
Fujitsu Siemens Amilo D 7850 (Computer 2)	Intel Pentium 4 3.0	1	ATI Mobility Radeon 9200 (64MB)	Windows XP SP2
Apple MacBook 1,1 (Computer 3)	Intel Core Duo 2.0	2	Intel GMA 950 (64MB)	Mac OSX 10.6 (Snow Leopard)
Acer Aspire 5740G (Computer 4)	Intel Core i5520M 2.4	4	ATI Radeon HD 5650 (1GB)	Ubuntu 10.04 (Lucid Lynx)

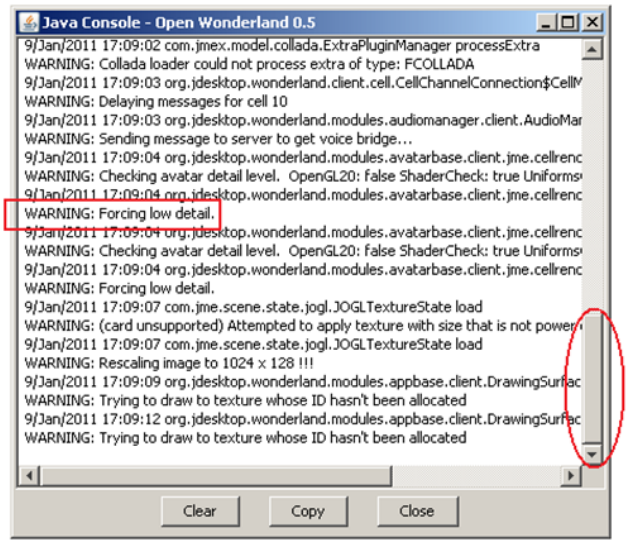

Figure 4. JWS status messages window

Concerning network location, computers 1, 2 and 3 accessed the VL inside and outside the case study's local network. When it is accessed from the outside, the 3D objects response delay is longer (but lower than 3 seconds) than when accessing inside the network. When a user accesses the WS in its network there are no nodes between them, in opposition to outside accesses. In data transmission, these network intermediate nodes always cause a little delay, which is almost undetectable in normal conditions. Otherwise, this can be a problem if a delay increases a little bit in one or more nodes. A series of small delays may become a long one, which then becomes highly detectable.

B. *Multi-user Scenario*

The VM features are used in this test scenario. The goal is to test the measured values retrieved by the RM and the access mechanisms. This test was made by two users in computers 1 and 2, both inside and outside the case study's local network. This test includes the next three situations:

a) VM being used and locked to other users;

b) The user who controls the VM leaves the VL world with the device on;

c) VM using after VL loading with the VM on.

Users *david* and *oliveira* access the VL world in situation a), as shown in Fig. 5.

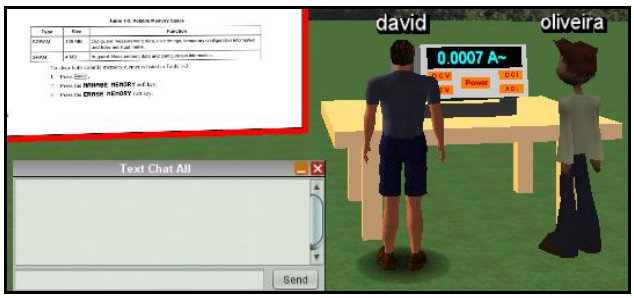

Figure 5. VL with two users

In first, user *david* turns the VM on and checks the measured values according to each button feature. This user also tries pushing the buttons from a distance close or far enough. If his avatar is close to the VM, its buttons should work. Otherwise, the buttons should not work. While this user is testing the VM main features, user *oliveira* should try to push the buttons in order to check if the device responds. It should not react because the VM is locked to user *david*, who turned it on first. After this try, user *oliveira* should ask for the VM control to *david*, using the text chat window (low left corner in Fig. 5). Then, user *david* releases the VM control by turning it off and *oliveira* should turn the device on to acquire the control. Subsequently these two users switch their test tasks. This test situation was successful as user *david* could use all VM features just when his avatar was close enough to the device. User *oliveira* could saw all *david's* measured values and actions over the VM. He was not able to use the device, because its use was locked to *david*. When this user turned the device off, user *oliveira* could use it by turning it on. In this case *david* was not able to use the device.

The situation b) test was successful as user *oliveira* left the VL with the VM on/locked, and *david* was able to reacquire the control by pushing any button. This means the device is never left locked if the controlling user leaves the virtual world.

If the VM is left on, it should be usable by any user, as its control is transferable and the last measured value should be in its display. This is the c) test situation, and was proved successfully, by user *oliveira*, because he could use the VM previously locked to *david* and saw *david*'s last measured value in the device display, after the VL loading. The VL was empty when user *oliveira* accessed it. The VL response time delay was longer (but below 3 seconds) for both users when these tests were done outside the case study's local network.

C. *Intercontinental Access*

In this scenario, the VL intercontinental access is tested. Usually, the further a place is, the more intermediate nodes in data transmission across the Internet it implies. This test was done by a Brazilian researcher, who accessed the VL from his country. He was represented by user *juarez*. The success criteria are the VL loading in less than or 35 seconds and the VM response delay should be 5 seconds or less. Any duration longer than these times makes the case study using inappropriate, and the test is considered unsuccessful.

This test was successful for user *juarez* as he could load the VL in less than 35 seconds and got measured values using the VM in less than 5 seconds. It was noticed a longer VL loading time in the user's first VL access than the following ones. The reason is because the JWS application has to download some files required for the VL first access, which is not necessary in the next ones, saving VL loading time. This user could use the VM with response delays below 5 seconds, after the VL loading.

D. *Initial Findings*

The times for the VL loading time and VM response delay, of 30 and 3 seconds, respectively, in subsection A test, and of 35 and 5 seconds, in subsection C, were defined according to the distance from where a user accesses the VL. In communication networks the number of intermediate nodes increases as the distance increases. As referred above, these nodes cause a little delay which may be detectable when they add in a series of small delays. When someone accesses the VL from a far place to the case study local network, as in the subsection C test scenario, it increases the delay detection probability. This detail was determinant to define duration times higher in the subsection C test scenario than in that of subsection A. It was not noticed in subsection A scenario any difference in the VL loading time between using the JWS application in a first time and in the following ones, as it happened in subsection C, where this detail was noticed due to the long distance data transmission delays.

The tests scope could be wider if more time was dedicated to these tasks or if automated testing tools were

used. Larger projects require a clear test policy to grant the appropriate running of their features.

V. Conclusion and Future work

This case study is a possible example of how remote laboratories can be accessed through 3D environments. During the case study development, the most important subjects to search were the VL building and the RM features. The first subject depends on the 3D environment features. It is important to search how to add content or configure its server in order to have a VL as similar as possible to a real one. In the second subject it is important to check if a measuring device can be remotely accessed in computer networks. If so, the next step is to check which network protocol or protocols are used to access the device. The most important developing phase was the communication between the RM and the VM implementation, provided by one of the OW most important features: its developing programming language. Most of the additional content to OW is Java programmed, whereas this language has additional code libraries, which provide functions for both appearance drawing and network protocols features programming.

One of the greatest challenges to OW is to improve its reliability, as this 3D environment is relatively recent.

The case study proved wide enough, as it required studying different domains like VL built, 3D and network protocols programming. It could include more features such as: more electric variables, an electric circuit linkage to VM virtual reproduction, the user pushing a button animation, or a VL appearance more similar to a real one, etc. The methodology used in this case study could certainly be the basis for building other virtual devices to have a complete virtual electronics remote lab.

Finally, using a device in a remote lab, accessible through a 3D environment, always depends on the device status and how it is physically linked to the electric circuit.

References

[1] What is a Virtual World? Virtual Worlds Review [online] [Accessed 2009-10-15]. Available at: http://www.virtualworldsreview.com/info/whatis.shtml.

[2] García-Zubia, J.; M. Castro, M.; Orduña, P. et al. SecondLab: A Remote Laboratory under Second Life, IEEE Global Engineering Education Conference, Madrid, Spain, April 2010 [EDUCON'10].

[3] Marcelino, R.; Silva, J.B.; Alves, G.R. et al. An extended immersive learning environment for solid mechanics theory and demonstration(s), International Conference on Remote Engineering & Virtual Instrumentation, Stockholm, Sweden, June 2010 [REV 2010].

[4] Scheucher, T.; Bailey, P.; Gült, C. et al. Collaborative Virtual 3D Environment for Internet-accessible Physics Experiments, June 2009 [REV 2009].

[5] VEL 2.0 - Virtual Enterprise Lab. Lucerne University of Applied Sciences and Arts [*online*] [Accessed 2011-01-25]. Available at: http://virtual.enterpriselab.ch/en/.

[6] Gandini, C. A laboratory of virtual veterinary. Interview held on the 3D world of Vetunimi [*online*] [Accessed 2011-01-25]. May, 2001. Available at: http://www.learnholistically.it/projects/3dfantasia/1issue/vetunimi_en.htm.

[7] Open Wonderland FAQ. Open Wonderland [online] [Accessed 2009-10-21]. Available at: http://openwonderland.org/about/faq.

[8] Open Wonderland v0.5: Download, Configure, Build and Run from the Source Code. Google Code [online] [Accessed 2010-03-06]. Available at: http://code.google.com/p/openwonderland/wiki/DownloadBuildSource05.

[9] Slott, J.; Kaplan, J. Open Wonderland v0.5: Launching Clients using Java Web Start. Google Code [online] [Accessed 2010-03-07]. Available at: http://code.google.com/p/openwonderland/wiki/WebStart05.

[10] Project Wonderland v0.5: Web-based Server Administration. Java.net [online] [Accessed 2010-03-06]. Available at: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandServerAdministration05

[11] Ganymed SSH-2 for Java. Cleondris [*online*] [Accessed 2010-09-25].Available at: http://www.cleondris.ch/opensource/ssh2/.

[12] Apache Commons Net. Apache Commons [*online*] [Accessed 2010-09-09].Available at: http://commons.apache.org/net/.

Authors

David Costa, is a MSC. Student at Instituto Politécnico do Porto - Escola Superior de Engenharia (IPP/ISEP), Porto, Portugal. (e-mail: 1020519@isep.ipp.pt).

Gustavo Alves, is a senior researcher at Instituto Politécnico do Porto - Escola Superior de Engenharia (IPP/ISEP), Porto, Portugal. (e-mail: gca@isep.ipp.pt).

Paulo Ferreira, is a senior researcher at Instituto Politécnico do Porto - Escola Superior de Engenharia (IPP/ISEP), Porto, Portugal. (e-mail: pdf@isep.ipp.pt).

Juarez Silva, is a senior researcher at Universidade Federal de Santa Catarina (UFSC), Araranguá, SC, Brazil. (e-mail: juarez.silva@ararangua.ufsc.br).