Using LabVIEW Remote Panel in Remote Laboratories: advantages and disadvantages

Pablo Orduña, Javier García-Zubia, Luis Rodriguez-Gil, Jaime Irurzun, Diego López-de-Ipiña
Deusto Institute of Technology – DeustoTech, University of Deusto
Bilbao, Spain
pablo.orduna@deusto.es

Fabricio Gazzola
Federal Institute of Santa Catarina (IFSC)
Florianópolis,Brazil
fabricio.gazzola@gmail.com

Abstract— **LabVIEW is a development environment from National Instruments, focused on the automation of processing and measuring equipment, and it is and has been for years a crucial tool in Educational Remote Laboratories. Three features are key for this success: a) a visual programming language called G, so developers don't need to work with traditional "text" programming languages, therefore achieving a wider range of experiment developers; b) a strong industrial support and c) the "Remote Panels", where the developer publishes the application automatically in a web browser. However, Remote Panels require a plug-in to run, not available for mobile devices neither all browsers in operating systems, and it clearly breaks with the ongoing web development trends, more interested in HTML5. This contribution shows how LabVIEW Remote Panels are used in Remote Laboratories, describing its inclusion in the WebLab-Deusto platform, and it describes the advantages and disadvantages of its use, comparing it with other existing approaches.**

Keywords- labview; remote laboratories; html5; e-learning

I. INTRODUCTION (HEADING 1)

An educational remote laboratory is a tool, which provides to students an access to real experiments through Internet. Nowadays remote labs are becoming a powerful didactic instrument in an engineering education. The Labshare project survey [1] shows that remote labs offer superior features in terms of flexibility, utilization, space saving, and safety issues.

Remote Laboratories have pursued further benefits. Complex experiments such as VISIR have been built [2], being so successful that VISIR has been deployed in 6 European universities at the time of this writing, along with other deployments scheduled in Asia, and other universities consuming it remotely. In terms of supported devices, the range has been improved [7], supporting m-learning with Remote Labs. The range of remote laboratories in general is so wide [6] that tools to index are required and efforts to create and maintain these indexes have been placed, such as Lab2go [8]. The amount of research lines within Remote Laboratories has been extended to include quality of learning [9], even existing workshops focused on the educational outcomes of remote laboratories. There are also efforts to consider the issue of group formation within the context of remote laboratories [10].

This increasing interest on Remote Laboratories has generated big efforts to share resources among different universities. The idea is that a provider university can share experiments that it is using with its students, so students of a consumer university can use them.

The MIT iLabs team is pioneer in this field, having shared batch experiments among different universities since 2004 [12], and interactive experiments since 2008 [5]. The iLabs are the most popular remote laboratories platform, being used in different countries among the five continents.

In the same line, the LabShare project in Australia is a publicly funded project focused on building a network of remote laboratories. As part of this project, the Sahara platform has been developed, and it has recently created the LabShare Institute as a not-for-profit organization that will be an independent service broker promoting, maintaining and even hosting remote laboratories.

In Europe, the LiLa project (Library of Labs) was created focused on providing the LiLa portal which manages both virtual and remote experiments.

Efforts towards the integration of these efforts through interoperability bridges have been placed, as detailed in [11] for iLabs with Sahara. In fact, the Global Online Laboratory Consortium (GOLC) was recently created to promote the development and sharing of remote laboratories, which includes members of most of the initiatives described, as well as members of WebLab-Deusto, described in section III.

The usage of LabVIEW in Remote Laboratories is wide spread. It has been used as an internal tool for data acquisition and interaction with hardware in different environments, such as in the VISIR project [2], where the "instrumentation server" is fully developed in LabVIEW, but the user interacts with an Adobe Flash application connected to a C++ server ("measurement server"), as well as in similar environments [3].

However, a key feature of LabVIEW for Remote Laboratories is the support for "Remote Panels". With this feature, experts on LabVIEW can create an intuitive user interface for instrumentation and publish it to the web.

LabVIEW comes with a web server that automatically deploys the application, and in the client side the user will be able to see the user interface. This makes LabVIEW a suitable and efficient tool for Remote Laboratory developers, and this feature has frequently been used in the literature, used in major Remote Laboratories as LabShare Sahara [4] and MIT iLabs [5].

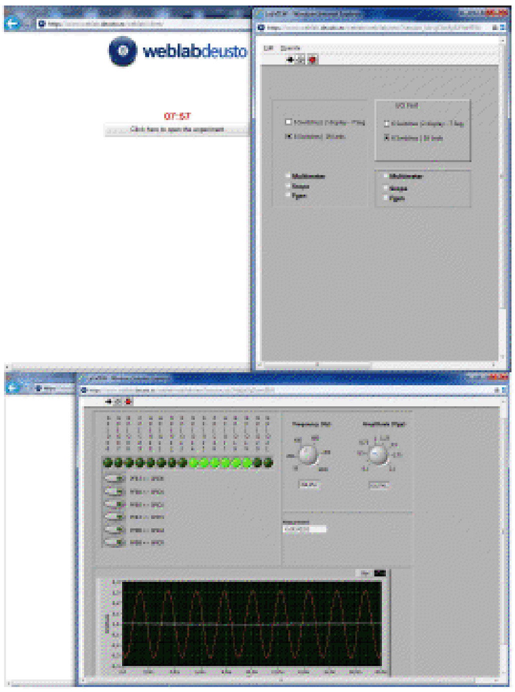

Figure 1 LabVIEW application running in WebLab-Deusto

However, the use of this tool not only requires the user to install a plug-in in her web browser, but also introduces some limitations in terms of security (limited authentication), deployment (no support for HTTP proxies, firewalls restrictions), latency (user might see data that has not yet been sent), scalability (for duplicating the experiment in order to balance the load among different rigs) or administration.

The remainder paper is structured as follows: Section II analyses the advantages and disadvantages of this technology, comparing it with other approaches, and Section III shows how LabView Remote Panels were supported in WebLab-Deusto. Finally, Section IV presents the conclusions and future work.

II. Analysing LabView Remote Panels for Remote Laboratories

Remote Panels is a technology provided by National Instruments LabVIEW that makes it possible to publish a locally designed user interface as the one shown in Figure 2.

The development of the user interface in LabVIEW is very simple and powerful, since common components such as switches, buttons, text areas, are available and the user interface is built in a drag and drop basis. The components are mapped to boxes in the programming area, which is also built in a drag and drop basis, with special boxes for loops, strings management, and of course, hardware management. Creating an application that interacts with certain hardware and does some management through a simple user interface becomes an easier task thanks to LabVIEW.

This user interface in LabVIEW by default is local. It can even be exported to an application file that will run in a Microsoft Windows environment that has installed the LabVIEW runtime.

However, with the introduction of the LabVIEW Remote Panels, it became possible to automatically publish the same user interface to a web browser. The process becomes very simple, given that LabVIEW comes with its own web server. The developer will select to publish the application and any remote user will see the panel as if he was in front of the local computer.

While LabVIEW loads a web server and the user accesses the application through a web browser (such as Microsoft Internet Explorer or Mozilla Firefox), it internally installs and runs a plug-in that uses the LabVIEW runtime. This plug-in, developed by National Instruments, is not available for all the web browsers. Nowadays it is particularly important that it will not work in mobile devices (such as mobile phones or tablets as iPad).

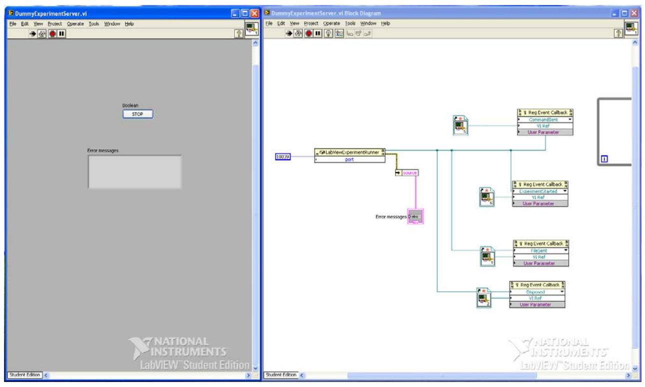

Figure 2 LabVIEW application. User interface in the left window, G programming language in the right window.

When the plug-in loads, it establishes a new TCP connection to the LabVIEW server. Since the transport is a plain TCP socket instead of the standard HTTP protocol, it can not be proxied through a web proxy. Fortunately, this connection is established using the same port as the LabVIEW web server, so no additional firewall management is required. However, using sockets instead of HTTP makes it impossible to deploy the LabVIEW in a server that is running any other web server (such as Apache or IIS) with other web applications, unless a different, not-standard port is used, and therefore dealing with firewall issues. It also makes it difficult to encrypt the communications using HTTPS as it can be easily done with a web service developed on top of HTTP.

The LabVIEW Remote Panels support optional authentication. The server defines a mechanism to establish which usernames have which passwords, and only those users will be able to access the system. Furthermore, it is possible to define if they have access in a read only way or in a read write way.

All these features altogether make it a suitable platform for quickly developing and deploying Remote Laboratories. However, it comes with clear drawbacks that laboratory developers must take into account.

A. Advantages of Remote Panels

The most noticeable advantages of LabVIEW Remote Panels are:

- Remote Panels an integral solution: it fits perfectly with all the tools already provided in LabVIEW to access, measure and interact with hardware. Whatever works in LabVIEW can be published in the web in a matter of seconds or minutes.
- Remote Panels are supported by National Instruments, providing industrial help.
- The learning curve for any developer who already knows LabVIEW is extremely low. The gap between the web development and LabVIEW development is completely covered. This includes communications, authentication and user interface building.
- The learning curve for anyone is very low, given the simplicity of LabVIEW. LabVIEW ignores the complexity of “text” based programming languages by providing the visual G programming language.
- The amount of potential Remote Laboratory developers increases significantly due to the learning curve. Anyone with LabVIEW knowledge can create a limited Remote Panel. As detailed in Section III, altogether with a Remote Laboratory management system such as WebLab-Deusto, the developer does not need to handle other issues such as authorization, administration, user tracking or scheduling.

B. Disadvantages of Remote panels

The main disadvantages of the use of Remote Panels are:

- The student is required to have LabVIEW runtime installed. While this can be an perfectly affordable requirement in his personal computer, it can become a problem if the student tries to use the system in a computer where he does not have administration permissions, such as a computer room in the university or a Cyber Cafe.
- The LabVIEW Runtime is not available for all platforms. While covering Microsoft Windows could be enough in most cases, it fails to achieve the ongoing trends of using mobile devices (including tablets such as iPad or Samsung Galaxy Tab) more often than the computer. Implementing m-learning with a Remote Laboratory based on the LabVIEW Runtime is plainly impossible.
- The communications are not encrypted, and they are hard to encrypt. The fact that it does not use HTTPS makes it difficult to create a tunnel that the web browser can understand and make it possible to send secure messages to the server. Depending on how affordable is the risk of malicious users breaking the remote hardware, it might become a problem.
- The communications are based on a TCP socket. This makes the deployment difficult when trying to support other web platforms (such as Apache or IIS). If it is required to have two servers in the same machine, a non-standard port must be opened, both by the IT services of the host university and by the client system (which might not be even possible if the user is in a foreign university).

C. Comparison with other approaches

It is possible to develop Remote Laboratories in other technologies. There are Remote Laboratories built on AJAX (Asynchronous JavaScript And XML, supported by all the web browsers) , on Java Applets, or Adobe Flash [2].

The use of these web based technologies makes it possible to achieve a wider range of devices (such as mobile devices), and to create less restrictions in the deployment of the Remote Laboratory, which is desirable especially if the student is not at home but somewhere else where he can not manage the firewall permissions (such as a public network in a hotel, a university, a bus, etc).

In fact, National Instruments itself has created another tool to publish web interfaces called NI LabVIEW Web UI builder, focused on decoupling the final user interface from the LabVIEW runtime. This solution is a step further in making it possible to develop completely web based Remote Laboratories using LabVIEW. The resulting solution requires Microsoft Silverlight, which is a competitor of Adobe Flash and Java Applets. While this solution is closer to the web trends (requires existing plug-ins instead of creating and maintaining one), it is not available for Linux neither for mobile devices. Furthermore, nowadays, with the move to HTML5 as a technology available in multiple mobile devices as well as in most web browsers, the use of plug-ins becomes unsuitable.

Finally, some Remote Laboratory developers have used LabVIEW successfully without using Remote Panels. The VISIR project, as stated in the introduction, internally uses LabVIEW but it presents a cross platform Adobe Flash application as a client, and the communications are based on HTTP. Therefore, they encapsulate the advantages of LabVIEW in a software solution that will work through different web browsers and operating systems, and even in Android. Other Remote Laboratories have chosen to use VNC to access the LabVIEW panel for other reasons, such as the difficulty of integration the LabVIEW runtime in other environments such as Open WonderLand [13].

III. INTEGRATION IN WEBLAB-DEUSTO

A. *WebLab-Deusto*

WebLab-Deusto [1] is a Remote Laboratory management system. The aim of WebLab-Deusto is to support Remote Laboratory developers, teachers and users, by providing them a tool on top of which they can develop new laboratories, manage them, and use them easily.

The clearest way to show what is WebLab-Deusto is to use an example. There is one teacher who desires to create a Remote Laboratory. He is an expert on FPGA systems (which are basically programmable devices), and knows very well how to setup a FPGA for educational purposes, how to evaluate the students, and how to create all the required educational material. He has some computing skills, so he can automate the process of sending a program to the FPGA through a USB port with a certain tool. Furthermore, he can write the web user interface of the laboratory, using Java applets, Flash, JavaScript or other technology.

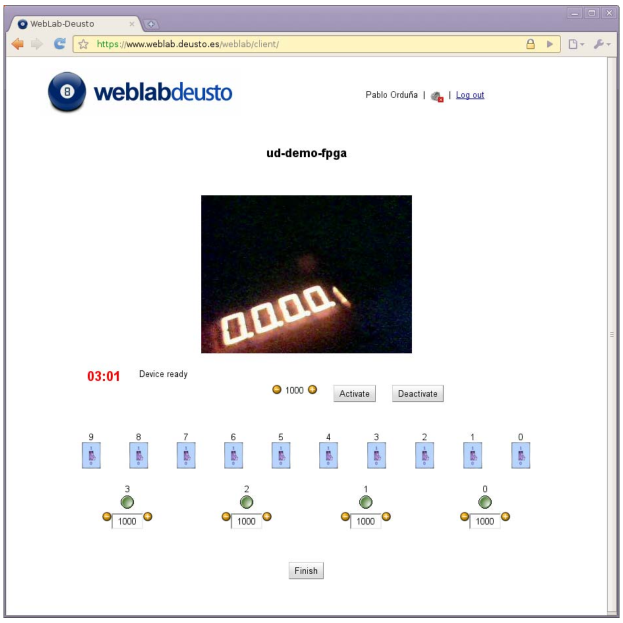

Figure 3 A FPGA laboratory built on top of WebLab-Deusto

Whenever he finishes that work, which completely depends on the system he is creating (FPGAs in this case), new features will be required to deploy it with the students of the class he is teaching. He will need to manage authentication (providing a username and a password). He will have to manage the scheduling (creating a queue or a calendar). He will at some point need to create administration tools, to add new users, remove privileges to others, check what is being done, etc. He should manage security in the communications (encrypting the information sent to the server, checking the firewall policies, etc.). He will want to be able to see what the users did (who accessed the system and when), and even he may want to make a further analysis of the FPGA laboratory he has done (where did the users enter from, their home or from the university? Was it during the classes, or the day before?).

However, all these features are common to any Remote Laboratory. It does not matter if the laboratory is using FPGAs, CPLDs, PIC, managing robots, controlling electronics or calling LabVIEW routines. All the management features will be the same, and they can be reused. This is the point where WebLab-Deusto comes in, as well as similar tools such as LabShare Sahara [2] or iLabs [3]. WebLab-Deusto focuses on providing these reusable features, dealing with the challenges of a university-level software platform.

For instance, let us take the feature of authenticating the user. At this moment, WebLab-Deusto provides five different ways to do so. It provides a simple database on which to store a password –a hash of the password- for a given user. But given that a Remote Laboratory is a service provided by universities, WebLab-Deusto also supports LDAP, which is a standard protocol used in companies to integrate the credentials among different applications. Since it is common to have a LDAP server in the universities, students can log in WebLab-Deusto using their university credentials. Furthermore, it supports OpenID, which is a single sign on protocol that lets students of one university log in another university with credentials of their original university. Given the increasing interest on Remote Laboratories and social networks, WebLab-Deusto accounts can be created or linked with Facebook accounts, so students can use WebLab-Deusto inside Facebook as yet another authentication system. Finally, it supports authentication based on IP addresses, in order to enable Learning Management Systems (such as Moodle or .LRN) to act in the name of certain users easily.

Authentication is an example of how a common feature can be implemented to support different scenarios in a reusable way for different types of laboratories. WebLab-Deusto provides complex scheduling mechanisms based on queues, a communications management mechanism that ensures the security of the integrated laboratories, a web administration panel, as well as command-line administration tools, and a common general interface for all the laboratories deployed on it, reducing the learning curve. These features are automatically available for any laboratory developed using WebLab-Deusto.

B. *Integrating laboratories in WebLab-Deusto*

A key aspect of WebLab-Deusto is the way it supports the integration of new and existing laboratories. Given the variety of technologies used in Remote Laboratories [6], it is important to support a wide range of technologies. WebLab-Deusto provides server-side programmatic libraries for Java, .NET, Python, C, C++ and LabVIEW, as well as client-side programmatic libraries for Java applets, Flash and JavaScript.

The usage of these programmatic libraries is very simple. The developer basically calls some methods to send a command from the client to the server, and WebLab-Deusto guarantees that the command is sent securely through the different layers, and it also guarantees that it has been stored

[1] http://www.weblab.deusto.es/

[2] http://www.labshare.edu.au/

[3] http://ilab.mit.edu/wiki/

for user tracking. Therefore, the developer does not need to manage authentication, authorization, encryption, etc.

A clear example of integration of a Remote Laboratory with WebLab-Deusto is the integration of the VISIR project [2]. The VISIR client is developed in Adobe Flash. Since WebLab-Deusto provides a library for Adobe Flash, and given the modular design of VISIR, it was possible to replace the communications layer by the WebLab-Deusto library. Automatically, the VISIR project was accessible from WebLab-Deusto. With it, VISIR students is enriched so it can be accessed from Facebook, and the user tracking that teachers can do is more fine-grained than the original, given that all the commands exchanged are stored in a database in WebLab-Deusto and they are accessible by the teachers with the administration panels provided by WebLab-Deusto.

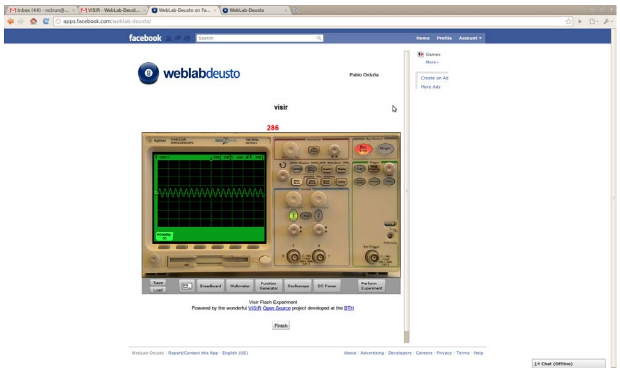
facebook
weblabdeusto
Pablo Orduña
visir
286
Visir Flash Experiment
Powered by the wonderful VISIR Open Source project developed at the BTH
Finish

Figure 4 VISIR experiment running in WebLab-Deusto, automatically accessible from Facebook

C. *LabVIEW Remote Panels in WebLab-Deusto*

With the growing user base of LabVIEW Remote Panels, its integration in WebLab-Deusto was considered an important interest. The target of the project was to support Remote Laboratory developers to create LabVIEW Remote Panels projects inside WebLab-Deusto, benefitting of the additional features provided by WebLab-Deusto.

In order to do so, a file-based protocol was established between a LabVIEW generic project and a new WebLab-Deusto server. Within this protocol, the LabVIEW project will store in the file certain information detailing messages meaning "load C:\path\file.vi", and the LabVIEW project responding "loaded" whenever it finishes loading it.

One of the main problems was that the authentication management. At application level, it is possible to require a user to manually provide a token provided by the platform. However, once a user logs in, any user with the URL might be able to interact with the system. In order to avoid this problem, LabVIEW provides a basic HTTP authentication mechanism, relying on an external file with usernames and passwords. It was not possible to programmatically modify this file, given that the hash algorithm used for the passwords was not a standard one. It was also not possible to rely on this file, since it would not be be possible to use externals authentication systems such as OpenID or the LDAP system, and it would require managing authentication at two places.

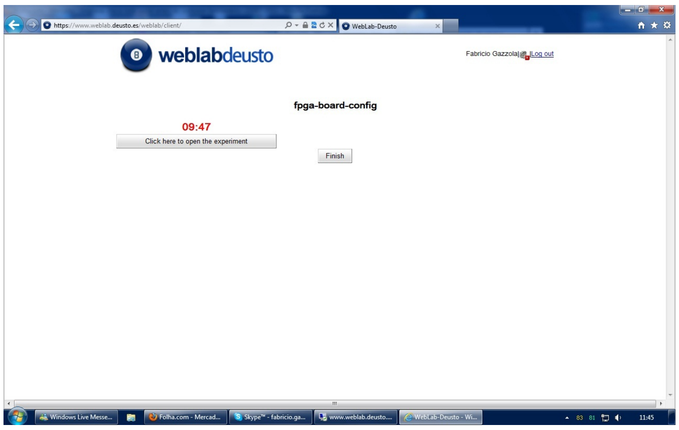

Figure 5 WebLab-Deusto initial LabVIEW user interface, without opening the LabVIEW Remote Panel

Given that the communication starts with HTTP, another approach was to establish a HTTP proxy between the client and the server, managing the authentication at the proxy. However, whenever the Remote Panel is loaded and starts, it does not use HTTP anymore but a TCP/IP socket that can not be used through any HTTP proxy. Furthermore, the communication sent through this socket does not contain any token that WebLab-Deusto can use to validate that the user is who claims to be if a socket proxy was established.

The final approach taken in WebLab-Deusto is to create a complete copy of the project to be loaded with a random token in the name (e.g. myproject_asdf13414fd). This token is sent through a secure communication to the student's browser, which will open it in a new window with this token, and therefore it will be able to access. Whenever the student is finished, WebLab-Deusto deletes the copy of the project and sends a message to stop the current connection to the LabVIEW generic project, so the student will not be able to access the system after the assigned time has passed.

The result is that the student, once logged in, selects the particular experiment if he has privileges to use it. Then, the user waits in the queue until he is granted the access to the real equipment, and he will have access to the window showed in Figure 5. Whenever the student clicks on the button, a new window is opened, accessing the real remote equipment, as showed in Figure 6.

While WebLab-Deusto itself supports mobile devices, and can be run in any standards-compliant web browser without requiring any plug-in, laboratories developed with LabVIEW Remote Panels will inherit the restrictions already detailed of this technology. Therefore, those particular laboratories will not run in mobile devices, neither in operating systems where LabVIEW is not available. However, these laboratories will benefit from the common features provided by WebLab-Deusto, such as not requiring to develop a scheduling mechanism, neither handling security, or developing administration tools that already come in with WebLab-Deusto.

All the developed code is open source and it is available in the public repository of WebLab-Deusto [4].

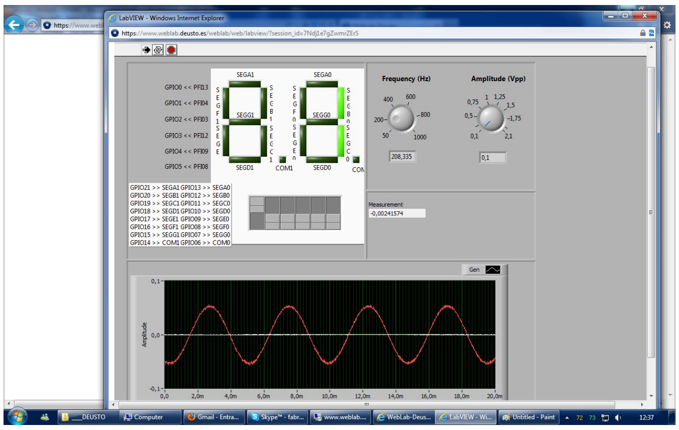

Figure 6 LabVIEW Remote Panel running inside WebLab-Deusto, in a new window

IV. Conclusions and Future Work

The contribution has presented an analysis of LabVIEW Remote Panels in Remote Laboratories, detailing its advantages and disadvantages. While it is a highly valuable tool and it is indeed widely used in the field, its use must be handled with care. The laboratory developer must be aware of the limitations it is imposing to students in terms of security, not making it available to certain operating systems (such as those running in tablets –iOS, Android-), and administration problems. The contribution also shows other approaches to deal with this problem, including LabVIEW Web UI Builder.

Finally, the contribution shows the integration of LabVIEW developed laboratories in WebLab-Deusto, obtaining all the benefits that WebLab-Deusto provides: authentication, authorization, administration tools, scheduling, user tracking, etc.

Related future work within the WebLab-Deusto project includes supporting LabVIEW Web UI Builder in order to let laboratory developers to create experiments using LabVIEW Web UI Builder. WebLab-Deusto already supports a set of technologies (Java applets, Flash applications, LabVIEW Remote Panels), so it should be possible to integrate Microsoft Silverlight and on top of it support LabVIEW Web UI Builder, but no work on this line has been started yet within WebLab-Deusto.

References

[1] Kostulski, T., Murray, S., (2010). "The National Engineering Laboratory Survey". Labshare Project. December 2010

[2] Gustavsson, I., Zackrisson, J., Håkansson, L., Claesson,I., Lagö. T.L., "The VISIR project – an Open Source Software Initiative for Distributed Online Laboratories", in Proceedings of Remote Engineering & Virtual Instrumentation Conference (REV), Porto, Portugal, June 25 – 27, 2007.

[3] U. Hernandez, J. Garcia-Zubia (2011). "A Remote and Reconfigurable Analog Electronics Laboratory based on IVI an LXI Technologies". Proceedings of the REV 2011 conference.

[4] R. Sarukkalige, E. Lindsay, A. Faisal. Laboratory demonstrators' perceptions of the remote laboratory implementation of a fluid mechanics laboratory. Proceedings of the 2010 AaeE Conference, Sydney.

[5] Harward, V.J.; del Alamo, J.A.; Lerman, S.R.; Bailey, P.H.; Carpenter, J.; DeLong, K.; Felknor, C.; Hardison, J.; Harrison, B.; Jabbour, I.; Long, P.D.; Tingting Mao; Naamani, L.; Northridge, J.; Schulz, M.; Talavera, D.; Varadharajan, C.; Shaomin Wang; Yehia, K.; Zbib, R.; Zych, D., "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.

[6] C. Gravier et al. State of the art about remote laboratories paradigms-foundations of ongoing mutations. International Journal of Online Engineering (iJOE). Vol 4, N. 1. 2008.

[7] P. Orduña et al. "Enabling mobile access to Remote Laboratories". IEEE EDUCON 2011 (ISBN: 978-1-61284-641-5). Amman, Jordan, April 2011.

[8] D. Garbi-Zutin, M. Auer, C. Maier, M. Niederstatter. Lab2go – a repository to locate educational online laboratories. Proceedings of the IEEE Education Engineering Conference (EDUCON) 2010. April 2010. Madrid, Spain.

[9] C. Gravier, J. Fayolle. Quality of learning: using a semantic web approach to enhance learner control during collaborative remote laboratories. International Journal of Innovation and Learning, vol. 6. Pp. 606-624. 2009.

[10] A. Mujkanovic, D. Lowe. Policy-Based Remote Laboratory Multi-User Access Management. Proceedings of the REV (Remote Engineering and Virtual Instrumentation) 2010. Stockholm, Sweden.

[11] H. Yeung et al., "Interoperability of Remote Laboratories Systems," International Journal of Online Engineering (iJOE). Vol 6. Special issue REV2010.

[12] J. Harward, et al. "iLabs: A scalable architecture for sharing online laboratories". International Conference of Engineering Education. 2004. Gainesville, Florida. October 16-21, 2004.

[13] Scheucher, T. and Bailey, P.H. and Gütl, C. and Harward, V.J. Collaborative virtual 3d environment for internet-accessible physics experiments. Proceedings of the International Conference of Remote Engineering and Virtual Instrumentaion. 2009

[4] http://code.google.com/p/weblabdeusto/