Advanced integration of OpenLabs VISIR (Virtual Instrument Systems in Reality) with Weblab-Deusto

L. Rodriguez-Gil[1], P. Orduña[2], J. García-Zubia[1], D. López-de-Ipiña[2]
[1] Faculty of Engineering, University of Deusto, Bilbao, Spain
[2] DeustoTech – Deusto Institute of Technology, University of Deusto, Bilbao, Spain

***Abstract*—During the last years, VISIR (Virtual Instrument Systems in Reality) has proved itself a useful tool for electronics remote experimentation, having been deployed in several different universities. As a domain-specific remote laboratory, VISIR offers those features which are required for its stand-alone usage, such as authentication, scheduling, user management, etc. Though for certain purposes this may be adequate, often it is more appropriate to offer VISIR as one kind of experiment among many, under a generic remote laboratories framework, such as WebLab-Deusto, MIT iLabs or Labshare Sahara. These frameworks provide integrated access to several different kinds of experiments, such as electronics, robotics, etc. Through this integration, a smooth experience can be provided to the user, and VISIR can benefit from all the functionality that the generic framework provides (common authentication, load-balancing, scheduling, etc). Efforts are currently being made to integrate VISIR with various laboratories. In this paper, we describe what the integration of VISIR with Weblab-Deusto involves; how certain VISIR-specific functionalities that depended on its original framework were handled, and how through Weblab-Deusto VISIR can easily gain certain new features. Some of those are the integration with different environments such as Facebook, or with Learning Management Systems such as Moodle. Another feature is collaboration among VISIR users, which makes it possible to share a VISIR circuit in real time. Furthermore, through this association VISIR gains new possibilities, such as federation.**

***Index Terms*— remote-labs, integration, VISIR**

I. INTRODUCTION

Nowadays, Remote Laboratory Management Systems such as MIT iLabs [1], Labshare Sahara [2] or Weblab-Deusto [3] coexist with domain-specific ones such as VISIR (Virtual Instrument Systems in Reality).

The former aim to provide a powerful, reusable and scalable framework that can be shared among a number of very different experiments. For instance, besides VISIR itself, some experiments that are currently available for Weblab-Deusto are several robotics experiments, some electronics experiments with Pic microcontrollers and FPGA (Field Programmable Gate Array) devices, etc. These remote laboratory frameworks can typically be extended easily with new experiments, because they already provide most of the core functionality that a remote laboratory requires. Hence, the developers of new experiments only need to worry about implementing their experiment-specific code. User management, authentication, authorization, etc, and even more advanced features such as load-distribution, logging, user tracking, replication or federation is already provided by the framework.

On the other hand, domain-specific remote laboratories tend to focus on providing access to what we could consider a single, but very complex experiment. That is the case with VISIR.

Created in the BTH and used by several other universities, VISIR is a system which provides a remote electronics laboratory. This laboratory is meant to be as similar to a traditional electronic laboratory as possible. Thus, it lets users (generally students) graphically design a circuit through a web-based Adobe-Flash interface. Through a relays system, the VISIR hardware can then physically "build" an equivalent of that circuit. Real physical measurements can then be provided to the user through the realistic user interface that the web client provides, which includes tools such as oscilloscopes or multimeters, commonly found on local laboratories. The hardware is capable of rapidly switching circuits, transparently serving and providing real physical measurements for many users (and their circuits) at once. It is important to remark that those measurements are not simulations; the circuit is physically wired, and the measurements taken.

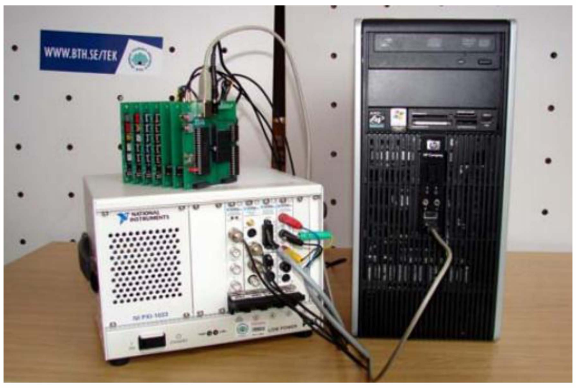

Figure 1. VISIR's hardware and equipment server. Through this means, circuits are physically built through the use of relays. [5]

VISIR, as a domain-specific laboratory, implements several layers of functionality that are actually common to all remote laboratories, such as user management, authentication and authorization, and the user interface required for these.

The provision of these basic layers is at times very convenient, because through them VISIR can be offered as a stand-alone solution that does not depend on sometimes more cumbersome Remote Laboratory Management Systems. However, because these basic layers are for them a requirement, but not really their focus, most often they lack certain advanced features that their more generic counterparts do provide, and they tend to present some significant issues and limitations.

Also, often the institution that wishes to host a VISIR instance will also be interested in offering several other experiments through a generic remote laboratory framework such as Weblab-Deusto. If VISIR is offered through its stand-alone framework, the users will have to access it through a completely different interface than every other experiment, with a different web-page, a different “login” screen, a different experiment queue, etc.

Frequently, even if no other experiments are offered, using a more advanced, specialized remote laboratory framework is desirable, due to the additional features that it offers. For instance, VISIR does not support authentication mechanisms using Single Sign On, neither federation, which are provided by these Remote Laboratory Management Systems.

In this context, efforts are being placed to support VISIR in other Remote Laboratory Management Systems, such as MIT iLabs [4].

In this paper, we will describe the steps that we took to integrate VISIR with such a framework (Weblab-Deusto), and the difficulties we overcame.

II. Related Work

A. *VISIR architecture*

VISIR [5], which is Open Source software, is made of four different, mostly independent components:

- Equipment server
- Measurement server
- Flash client
- Openlabs Web layer

The equipment server deals with the hardware and the circuit-wiring robot. The measurement server works with the equipment server to provide the real-time measurements that are meant to be provided to the client. The flash client, connected to the measurement server, provides the experiment user-interface, easily accessible through most browsers. Finally, the Openlabs Web provides the aforementioned basic remote experimentation layers, including the initial web-pages, user authentication and authorization, access to the different laboratories, etc. This layer also includes a database, used to store users, circuits and other information.

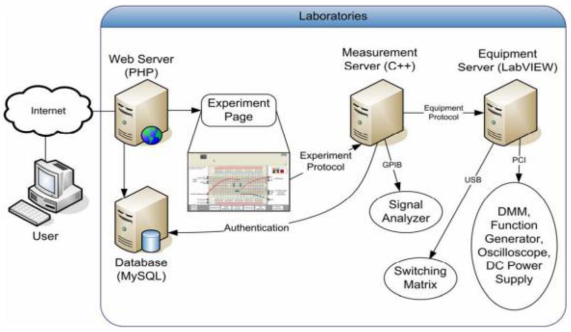

Figure 2. VISIR architecture. Organization of the online laboratories. [5]

Additionally, a few specific functionalities of the Flash client are also dependent on the Web layer.

It might be noteworthy that, as we will discuss in later sections of this paper, this layer is not really necessary when VISIR is to be integrated in a RLMS, and one key step of the integration process will be removing the dependency on it.

B. *WebLab-Deusto*

WebLab-Deusto is a Remote Laboratory Management System. As such, it provides the generic (authentication, user management, etc) and advanced (federation) features discussed in previous sections.

Through WebLab-Deusto students can access *experiments*. An example of an experiment is, for instance, a robot. Students can start the experiment and then, through the interface provided by the experiment (in this case, buttons and a video-stream) see the robot in real-time and move it around.

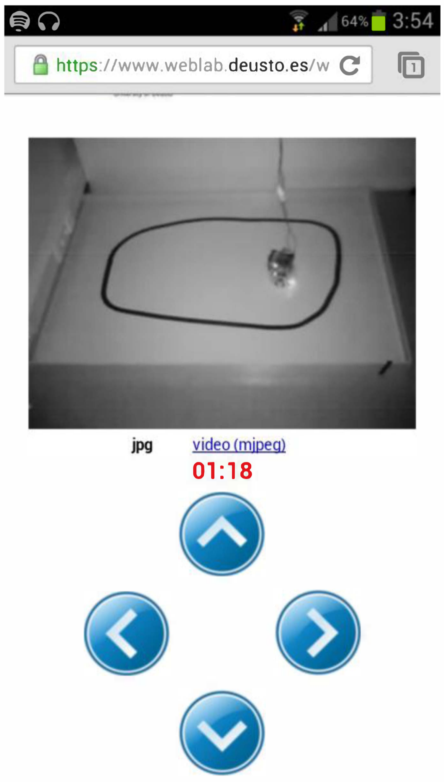

Figure 3. One of WebLab-Deusto robotics experiments (on a mobile phone, at night, video stream provided by an infrared camera).

In the general case there are essentially three main components to a WebLab-Deusto experiment.

First, an experiment needs the actual hardware behind it. Following the previous example, this hardware would be the robot (and, in practice, the hardware necessary to send the actual commands to the robot).

Second, an experiment needs a user interface, which we will refer to as the *experiment client*. This would be the web interface with the buttons and the video-stream to control the robot, and the logic behind it. It might be worthwhile to remark that each experiment has its own client, interface, and client-side logic, and they will thus tend to be very different from one another. In fact, WebLab-Deusto supports several different technologies for its clients, such as GWT (plain web), Adobe Flash or Java Applets, or even non-web-based technologies such as C# (.NET) or C++.

Third, an experiment needs a server-side component to define its specific logic and behavior. This is what we know as the *experiment server*. In the robot example, when the student requests the robot to move (through the *experiment client*) a message is sent to the *experiment server*. The experiment server will then process that message, and take the appropriate action. In this case, a lower-level message will be sent to the board that physically controls the robot, and it will then move.

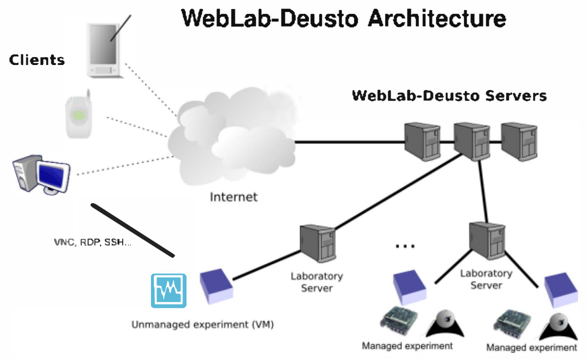

Figure 4. WebLab-Deusto architecture. Experiment servers are hosted in the Laboratory servers.

III. Approach

As explained in more detail in the previous sections, the VISIR software is made essentially of four different parts: The flash web client, the web interface, the measurement server, and the equipment server. We want to provide fully functional integration into Weblab-Deusto, so that we can leverage what such an integration offers. Nonetheless, at the same time we want to minimize modifications to VISIR itself. For this, we have had to consider several different approaches. We will discuss those approaches we have discarded first, and explain the one we have finally chosen afterwards.

A. *Shallow integration*

The first strategy we may consider could be to simply 'embed' the standard VISIR web interface within Weblab-Deusto, without further modifications. This could be done, for instance, by using an *iframe*. It would be particularly fast and straightforward to implement. However, it would not really meet our requirements. Users would indeed access VISIR through Weblab-Deusto, but they would still have to navigate through the VISIR web interface itself, and they would have to authenticate to both Weblab-Deusto and VISIR. Thus, the user experience would not be smooth at all, which was actually one of our main goals. Besides, from such a shallow integration, we would not get any of the advantages mentioned in previous sections (scalability, auditing, etc). We can hence discard this too simplistic approach.

B. *Authentication replacement*

As a more complicated variation of this, we could consider embedding only the standard VISIR flash client, and not the whole VISIR web interface. Though contrary to the previous approach, certain modifications to VISIR would be required, these modifications would be limited to the web interface. The flash client (and of course, the other two VISIR components) would remain untouched. Also, the changes required to the web layer would not be too many. Replacing the user authentication code (so that users need only authenticate once with Weblab-Deusto itself) would be the most significant change. Additional work would also be required for circuit listing and selection.

This approach would be significantly more effective than the previous. The experience would be seamless to the end-user, and we could at least to some extent leverage many of the advantages that Weblab-Deusto provides, such as scalability, load-balancing, scheduling or even other more specific features such as federation or automatic facebook and LMS integration. Unfortunately, certain limitations exist. First, even though we replaced the authentication scheme, the system is still fully dependent on the VISIR PHP web layer and on its database. This is somewhat inconvenient and hinders maintainability because most often remote laboratory frameworks such as Weblab-Deusto will have their own database and often will use different server technologies (and will thus be unwilling to depend on PHP). Second, though this could be made mostly transparent to the user, the VISIR client is not fully integrated with Weblab-Deusto, and does not work the way standard experiments do. The flash client communicates directly with the measurement server. Thus, Weblab-Deusto has no control over the actual experiment. Unlike with standard experiments, it is not possible to audit the actual activity of the user, or to access his or her results.

C. *Our approach (Deep integration)*

Due to the reasons exposed above, we have chosen a different approach, which is more complete though also slightly more complicated. In this approach, we will completely replace the VISIR web layer. Thus, we will not be dependent on PHP or on the VISIR database anymore. Also, we will route all VISIR traffic through Weblab-Deusto itself, in order to obtain full auditing capabilities and full control over the experiment. Though this does indeed require certain modifications to the VISIR client, thanks to the client's modular design those modifications are scarce.

D. *Summary*

As discussed in further detail in the previous section, several approaches were available. Out of them, we chose the one which probably involves the most work, but also the most features. We will next summarize advantages and characteristics of each approach through three tables.

The first table lists the three approaches that we have considered: Shallow integration, authentication-only integration, and deep integration.

TABLE I.
CONSIDERED APPROACHES

	Approach	Involves
A	Shallow	Embedding the whole, standard VISIR website.
B	Auth-only	Replacing the authentication procedure of VISIR (implemented in the VISIR web layer).
C	Deep (chosen)	Replacing the whole web layer and proxying all communication through the RLMS.

The second table specifies which features are supported by each approach, and which are not.

TABLE II.
SUPPORTED FEATURES

	A	B	C
Accessible through the RLMS [a]	Yes	Yes	Yes
Smooth user experience	No	Yes	Yes
Logging & auditing	No	No	Yes
Full experiment control [b]	No	No	Yes
Extended features [c]	No	No	Yes

[a] Remote Laboratory Management System. Weblab-Deusto, in our case.

[b] Ability to start and stop the experiment, thus being able to control the number of users and their use of the experiment, etc.

[c] Features such as supporting several different VISIR instances, control over the components definition file, circuit publishing, etc.

The third table lists which of the four VISIR components need to be modified to implement each approach.

TABLE III.
CHANGES REQUIRED TO VISIR (COMPLEXITY)

	A	B	C
Web layer (minor changes)	No	Yes	Yes
Web layer (major changes)	No	Yes	Yes
Flash client	No	No	Yes
Other components [a]	No	No	No

[a] Particularly, the equipment and measurement servers. None of the methods requires to changes to those.

We can easily appreciate an incremental pattern in both tables. The approaches can be ordered in terms of features and complexity, and then each one is in fact a subset of the next according to either criteria.

IV. IMPLEMENTATION

A. *Basic integration*

As previously discussed, we chose a powerful but relatively complicated approach for our integration, which required major changes to the web layer (it will be replaced altogether) and relatively minor changes to the VISIR Flash client.

We have divided the integration process in several stages. After this first stage is completed, authentication will be working through Weblab-Deusto, and all communication to VISIR will be proxied through the RLMS (and hence, all users, their commands and their results logged). All basic VISIR features will be working, except for some features which depend on the now-gone VISIR web-layer, and except for some additional features which are extensions to the original VISIR.

The VISIR client originally supports two different “transports”. These transports are the channels through which it can communicate with the measurement server. Their names are specifically TransportHTTP and TransportXMLSocket. As they suggest, the first one communicates through HTTP while the second communicates through sockets.

We have developed a new transport, TransportWeblab. Through this new transport, we accomplish the goal of routing all communication through Weblab-Deusto. Internally, this is achieved through the use of a Javascript interface that Weblab-Deusto provides.

Using this approach, we were able to get most VISIR functionalities working through Weblab-Deusto.

Additionally, this integration provides several advantages:

- Coherent user interface
- Load-balancing
- Logging & auditing capabilities
- Scheduling
- User management & authentication
- Control over the experiment & command interception capabilities

There are still, however, certain limitations. As we mentioned in previous sections, we have chosen to no longer have the VISIR Web layer (unless we installed it, which is not what we want). There are certain features of the VISIR flash client which depend on it. Hence, without further changes, those features will not be available anymore. An additional effort will be required to re-implement these features. Sometimes, in fact, we will even be able to improve and extend these features, or even add altogether new ones. These matters will be discussed in later sections. We will next discuss which features were no longer working due to their web-layer dependency, and the steps we took to resolve those issues.

B. *Circuit list*

In VISIR, a “circuit” is the current layout of the wires and components. It includes both a layout in the breadboard and the set of components that are available to the user.

The most noticeable feature which the original VISIR web layer provided is probably the ability to choose a circuit among a list.

This is a particularly useful feature, because by means of this system, teachers can accomplish two goals:

- Provide a starting point
- Limit the choice of components

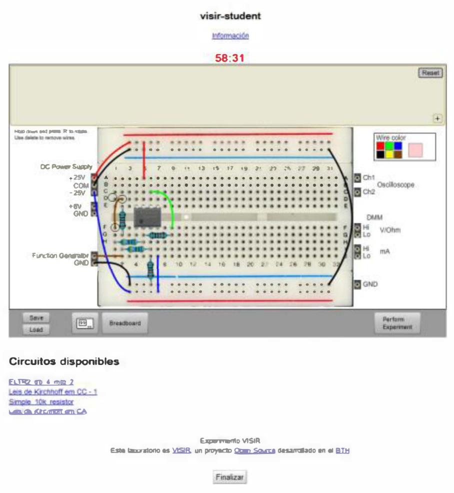

Figure 5. VISIR and the circuits list, on the lower left.

Though it would be possible for teachers to simply let users start with a blank breadboard, and to let them choose any component they wish, this is often inconvenient. It is often more appropriate to start with several components and wires already placed, so that the student can focus only on those parts relevant to his specific course or experiment, and not waste time building the circuit from scratch. Likewise, limiting the choice of components is often a good choice, because having the whole list available adds unneeded complexity. It is often more convenient to only expose the student to those components which are relevant to the specific course or experiment.

In the standard version of VISIR, this feature depends on the Web layer for the interface and logic, and on its database (which is where the circuits are stored). As explained in previous sections, when integrating VISIR in Weblab-Deusto we do not want to depend on the (php) Web layer, nor on its database.

This is therefore one of set of functionalities that we have had to reimplement.

The web-based graphical user interface is, essentially, a simple list, so there were no issues at that respect. The main issues were storing and defining the circuits, and loading these circuits upon selection. We can either embed the circuit within our VISIR "experiment" configuration, or just, in this same configuration, point the experiment to a folder, from which it will load all circuits. This is a particularly powerful approach. First, because Weblab-Deusto supports having any number of VISIR "experiments", each of which can thus have, very easily, a different configuration and set of available circuits. And second, because being able to simply load the experiments from a folder lets us, for instance, point the experiment to a shared Dropbox folder, which the teachers themselves can easily manage.

The circuits are thus "served" by our VISIR experiment, once requested remotely from the client upon clicking on the list. It is noteworthy that it is requested through exactly the same channel that the VISIR flash client uses to communicate with the VISIR experiment, and through it with the measurement server.

It is maybe noteworthy that we had to tackle certain issues for supporting on-line circuit reloading. In the original VISIR, the base circuit is specified through the "flashvars" parameter. This parameter, common to all Flash applications, is passed to the application on startup, on the HTML itself. Weblab-Deusto was originally not fully prepared for that kind of requirement, so we made certain modifications. It is now possible to dynamically reload the flash application upon request (in our case, with a new flashvar parameter, defining the chosen circuit). This is done transparently, and the CTransportWeblab is also re-initialized as required.

C. *Saving a circuit*

As explained in the section before, it is, once again, possible to select a circuit. However, yet another feature that depends upon the Web layer is the "save" button. Through the "save" button, the VISIR client originally carried out an HTTP POST request to a PHP script, which essentially generated a circuit file, ready for being downloaded by the user. To be able to support this feature (and the circuit loading one, which will be explained in later sections), we have implemented in Weblab-Deusto a proxy, capable of intercepting file requests. Thus, when the VISIR flash client requests a standard, static file such as "loader.swf", this interceptor will simply examine the request, read the file from disk, and provide the file, as a typical HTTP server would do. However, under other circumstances, this interceptor will be able to provide additional functionalities. These functionalities, at times, go beyond simply integrating VISIR features, and through it, VISIR can even be extended with new features.

Circuit saving is done in VISIR through a post request to the a "/save" folder, which was originally handled by a PHP script. These requests are now handled by our interceptor, which will do what the PHP file used to. That is, extract the circuit data, and provide it as an HTTP download to the requesting user.

D. *Loading a circuit*

Another feature that depends on the Web layer is the one provided by the "Load" button. This button lets the user choose a local circuit file, which can then be loaded. In the original VISIR, there is a set of PHP scripts which let the user upload the circuit, save that circuit as a temporary file to a temporaries folder, and then report to the Flash client the URL of that new temporary file containing the circuit. The Flash client is prepared to then automatically load that circuit. In Weblab-Deusto, we, once again, intercept that upload request. We then, too, save the file to a temporary folder. However, to gain additional safety and flexibility (which is in our case particularly important, because we can actually have several different instances of VISIR experiments), the URL that we provide to the client is not the real URL to the temporary file. Instead, it is an identifier. When the client then requests the file through that identifier, the interceptor, once again, intercepts the request, and is able to serve the right file, without actually disclosing the location of our temporary folder (and thus supporting, in fact, any number of temporary folders).

V. IMPROVEMENTS AND EXTENDED FEATURES

As mentioned before, through the Weblab-Deusto integration VISIR itself can actually gain new features.

Through the interceptor, for instance, we intercept "library.xml" requests in order to dynamically provide a library.xml depending on the experiment. Thus, Weblab-Deusto can host at the same time several VISIR experiment instances, each with a different library.xml.

Another very interesting feature, which will be explored in further detail in the future, is collaboration. Through Weblab-Deusto and the circuit-selection and reloading system, it is possible for one user to "publish" his current circuit. This circuit will then be temporarily stored in the experiment server's memory. Then, the other users which may be concurrently using VISIR can gain access to the experiment the user published, and load it in seconds, just as they would load a standard configured circuit.

It might be noteworthy that to make this possible, it was necessary to implement, in the VISIR Flash client itself, a new ActionScript method, SaveExperiment, which is analogue to the pre-existing LoadExperiment. While the later loads an experiment taking the XML string describing it as input, the former returns the string describing the current experiment. Thus, we gain access to the current circuit through JavaScript (and hence through WebLab).

VI. Conclusions

The approach we have chosen has taken a significant effort to implement. However, it has proven itself effective.

Teachers and students at the University of Deusto are already using VISIR through WebLab-Deusto.

Essentially, all features supported by the original VISIR are now supported as well, and a smooth experience is provided to the users, who can seamlessly access VISIR along with the other experiments that WebLab-Deusto provides.

Furthermore, these users are already enjoying some of the additional features (not supported by the original VISIR) that are now made possible through the integration.

Thanks to the new circuit listing and loading system, teachers are now easily managing the circuits list through Dropbox (a shared folder service).

Through the federation support that is provided by WebLab-Deusto, other institutions such as the Colegio Urdaneta school have gained access to Deusto's VISIR instance. (See Figure [3]).

Also, it is now possible to better control and schedule access to VISIR, as it is possible to limit concurrent access to a certain number of users, all while making use of WebLab-Deusto advanced scheduling and prioritizing capabilities.

Moreover, all this is done securely, as it is possible to log and to analyze VISIR usage, as usage information, including every command and result sent to and from VISIR, is now being recorded and stored in the WebLab-Deusto database.

Through WebLab-Deusto, it is now even possible to access VISIR from Facebook or through LMSs such as Moodle.

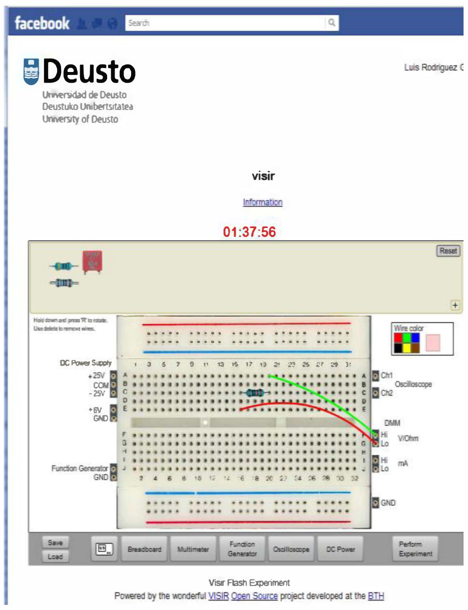

Figure 6. VISIR running in Facebook through Weblab-Deusto.

VII. Future Work

Circuit publishing is currently working experimentally. As of now, VISIR users can indeed share their current circuits with other users. However, this feature is in a very early stage and can't be used in a practical way yet. Particularly, there are many security, design and usability issues which are still left to tackle.

Efforts are already being dedicated (and will continue in the future) to improve these features, and to improve collaboration capabilities of Weblab-Deusto in general, and of VISIR through Weblab-Deusto in particular.

References

[1] V. J. Hardward, J. A. del Alamo, S. R. Lerman, P. H. Bailey, J. Carpenter, K. DeLong, C. Felknor, J. Hardison, B. Harrison, I. Jabbour *et al.*, "The ilab shared architecture: A web services infrastructure to build communities of internet accessible laboratories," *Proceedings of the IEEE*, vol. 96, no. 6, pp. 931-950, 2008.

[2] R. Sarukkalige, E. Lindsay, and A. Anwar, "Laboratory demonstrators perceptions of the remote laboratory implementation of a fluid mechanics laboratory," 2010.

[3] P. Orduna, J. Irurzun, L. Rodriguez-Gil, J. Garcia-Zubia, F. Gazzola, and D. Lopez-de Ipiña, "Adding new features to new and existing remote experiments through their integration in weblab-deusto," *International Journal of Online Engineering (iJOE)*, vol. 7, no. S2, pp. pp–33, 2011.

[4] D. Garbi Zutin, A VISIR Lab Server for the iLab Shared Architecture. International Journal of Online Engineering (iJOE). 7(5). 2011.

[5] I. Gustavsson, J. Zackrisson, L. Håkansson, I. Claesson and T. Lagö, "The VISIR project – an Open Source Software Initiative for Distributed Online Laboratories", Proceedings of the REV 2007 Conference, Porto, Portugal, June 25 – 27, 2007.

AUTHORS

L. Rodriguez-Gil is with the Faculty of Engineering, University of Deusto, Avda. Universidades 24, 48007 Bilbao, Spain (e-mail: luis.rodriguez@opendeusto.es).

P. Orduña, is with the Internet Unit of the Deusto Institute of Technology, DeustoTech, University of Deusto, Avda. Universidades 24, 48007 Bilbao, Spain (e-mail: pablo.orduna@deusto.es).

J. Garcia-Zubia is with the Faculty of Engineering, University of Deusto, Avda. Universidades 24, 48007 Bilbao, Spain (e-mail: zubia@deusto.es).

D. López-de-Ipiña, is with the Internet Unit of the Deusto Institute of Technology, DeustoTech, University of Deusto, Avda. Universidades 24, 48007 Bilbao, Spain (e-mail: dipina@deusto.es).