wCloud: automatic generation of WebLab-Deusto deployments in the Cloud

Pablo Orduña*, Aitor Gómez-Goiri*, Luis Rodriguez-Gil*, Javier Diego†, Diego López-de-Ipiña*, Javier Garcia-Zubia*
*DeustoTech - Deusto Institute of Technology
University of Deusto, Bilbao, Spain
Email: {pablo.orduna,aitor.gomez,luis.rodriguezgil,dipina,zubia}@deusto.es
†CGI
Madrid, Spain.
Email: javier.diego@cgi.com

***Abstract*—Educational remote laboratories are software and hardware tools that allow students to remotely access real equipment located in universities as if they were in a hands-on-lab session. Since most remote labs share certain management tasks (authentication, Learning Analytics, scheduling, etc.), software systems implementing them on top of which remote labs could be implemented were developed and called Remote Lab Management Systems (RLMS). A key feature provided by certain RLMSs is sharing a remote laboratory between two systems deployed in two institutions. This way, it becomes possible to have multiple RLMS instances (which are pure software) in a Cloud environment, customized for different schools or universities. Each school would have its own RLMS, with all the management features (e.g., managing its own students), and in the end, the RLMS would connect to the RLMS which has the physical equipment. The focus of this contribution is to detail how this "RLMS as a Service" is being implemented in the case of WebLab-Deusto as part of the mCloud project, from a technical point of view.**

I. Introduction

A remote laboratory is a software and hardware tool that allows students to remotely access real equipment located in the university. Users access this equipment as if they were in a traditional hands-on-lab session, but through the Internet. To show a clear example, Figure 1 shows a mobile low cost robot laboratory described in [1]. Students learn to program a Microchip PIC microcontroller, and they write the code at home, compile it with the proper tools, and then submit the binary file to a real robot through the Internet. Then, students can see how the robot performs with their program through the Internet (e.g., if it follows the black line according to the submitted program, etc.) in a real environment.

With time, several remote laboratory developers had to develop different remote labs of different nature. Instead of starting from scratch when developing these new remote laboratories, they started building software systems that could be reused among these different labs. This way, the development was splitted in two blocks: the laboratories code (e.g., the connection with the real equipment and the logic of the laboratory), and the management code (e.g., authentication, authorization, scheduling, user tracking mechanisms or administrative tools). These systems have been called Remote Laboratory Management Systems (RLMS). This way, a fair degree of shared development of remote laboratories is achieved: if one (or more than one) institution develops a RLMS, other institutions developing their remote laboratories on top of this RLMS will not need to develop those features again and will instead use that RLMS. A key feature of these systems is that they can share the managed laboratories to other systems in other universities.

So as to gain adoption, these RLMS could be installed in schools or universities which are not developing their own remote laboratories. For example, a secondary school could install their own RLMS in a server, and manage their own students, while using remote laboratories deployed in other universities. This way, there could be students, groups of students, teachers could decide which groups would have permissions on what laboratories, and finally teachers could track the usage of their students. Students and teachers would have a RLMS with the logo of their school.

However, deploying a RLMS is not a trivial issue for most schools, which might not have an IT services department, servers, or time or resources to deploy and maintain these RLMS. In this context, it is more suitable if RLMS developers provide a cloud approach where the RLMS is deployed automatically in the servers of the institution. This way, teachers would have the same benefits without requiring expensive resources.

The focus of this contribution is to explain how wCloud manages this with the WebLab-Deusto RLMS, using tools developed in the Spanish mCloud project.

II. Remote Laboratories Federations

This section introduces the concepts of Remote laboratories, Remote Laboratory Management Systems and remote laboratory federations.

A. Remote Laboratories

A remote laboratory is a software and hardware tool that allows students to remotely access real equipment located in the university. Users access this equipment as if they were in a traditional hands-on-lab session, but through the Internet. To show a clear example, Figure 1 shows a mobile low cost robot laboratory described in [1]. Students learn to program a Microchip PIC microcontroller, and they write the code at

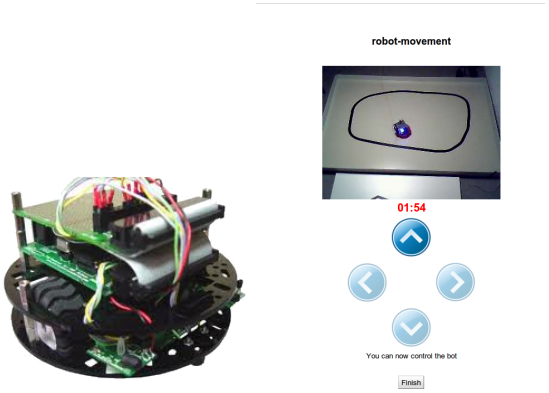

Fig. 1. Robot laboratory [1]. At the left, the mobile robot itself. At the right, the user interface once the program has been submitted.

home, compile it with the proper tools, and then submit the binary file to a real robot through the Internet. Then, students can see how the robot performs with their program through the Internet (e.g., if it follows the black line according to the submitted program, etc.) in a real environment.

In this line, there are many examples and classifications in the literature [2], [3]. Indeed, remote laboratories were born nearly two decades ago [4], [5], [6], and since then they have been adopted in multiple fields: chemistry [7], [8], physics [9], [10], electronics [11], [12], robotics [13], [14] and even nuclear reactor [15].

B. Remote Laboratory Management Systems

Every remote laboratory manages at least a subset of the following features: authentication, authorization, scheduling users to ensure exclusive accesses –typically through a queue or calendar-based booking–, user tracking and administration tools. These features are common to most remote laboratories, and are actually independent of the particular remote laboratory settings. For example, an authentication and queuing system is valid both for an electronics laboratory and for a chemistry laboratory.

For this reason, Remote Laboratory Management Systems (RLMSs) arose. These systems (e.g., MIT iLabs[1], WebLab-Deusto[2] or Labshare Sahara[3]) provide development toolkits for developing new remote laboratories, as well as management tools and common services (authentication, authorization, scheduling mechanisms). The main idea is that by adding a new feature to one of them (e.g., supporting LDAP, or LMSs), all the laboratories which are developed on top of them will support this feature automatically.

C. Federating Remote Laboratories

One of the features that RLMSs started supporting was federating their remote laboratories. For example, if two universities (*University A* and *University B*) install a particular RLMS, they support federation protocols so *University A* shares a laboratory with students of *University B* without knowing these students. The key here is that the provider university does not need to register particular students, but rather groups or simply universities. It is the consumer system who defines that a set of local users can access a particular laboratory of the provider system.

Therefore, the relationship between two federated entities is the following:

- The consumer system manages the authentication and authorization of its students.
- The provider system manages the scheduling and the access to the laboratories, storing what the users did.
- The consumer system will later ask for results to the provider system.
- In every moment, the provider system does not need to know anything related to the particular students.

These federation protocols have been used for fostering interoperability between RLMS [16]. These interoperable bridges between different systems can be enhanced if properties such as transitivity or federated load balance are provided [17].

III. RLMS as a Service

Through the federation protocols, the consumer RLMS does not need to have equipment connected and can become a purely software system. One school or university could have a RLMS deployed, and access laboratories deployed in other institutions. This way, the institution could be in charge of the students, groups of students, permissions of groups on foreign resources, or Learning Analytics tools [18] for instructors evaluating their students. Figure 2 shows two of the scenarios of Learning Analytics available to teachers using the system.

However, deploying a RLMS might not be convenient neither feasible in all the situations. For instance, schools might not have an IT department willing to download, install and maintain such server. Lecturers of certain universities might be interested in using a RLMS to access laboratories but might not have the skills, interest, resources or time to perform the whole process.

If RLMS providers created a system that enables external users to create institutional deployments in a Cloud basis, it would be much easier for the potential audience willing to use laboratories to adopt them. Lecturers would use it as a regular cloud service, and they would benefit from all the features provided by the RLMS: managing their students and permissions, renaming experiments or descriptions on the laboratory, and they would keep a detailed track of what users did on the final laboratory.

This concept could be compared to how Wordpress, the Open Source popular web solution for developing websites or blogs. Wordpress.org[4] provides the Open Source software and documentation on how to deploy it and manage it. Many corporate and personal blogs rely on this software, deployed in their servers and with no dependency on the original authors (except for updates, which are also free and Open Source).

[1] http://ilab.mit.edu
[2] http://weblab.deusto.es
[3] http://labshare-sahara.sf.net
[4] http://wordpress.org/

Fig. 2. Examples of Learning Analytics available to registered users

This model has created a community of people creating themes, plug-ins, tutorials and so on on how to work with this software. The initial release was published on 2003. However, to install the web application you need a server or hire a cloud environment and install it managing the databases, managing the network and providing maintenance. This is a problem for its adoption among those users who do not have the skills, time or resources to do so. To solve this, in 2005 Wordpress.com[5] was created to fill this problem. On Wordpress.com, any user can create a new blog, and start working on it from the very beginning, installing themes and plug-ins from a repository of audited themes. Wordpress.com, hosting this service, does not support advanced users changing some parts of the code, and it stores everything (users, uses, etc.) on their servers, but this solution is more convenient for many users than requiring everybody to manage their own system.

Relying on this concept of "RLMS as a Service", WebLab-Deployer [19] was developed as an ad hoc solution. It aims to provide the same benefits: WebLab-Deusto is an Open Source system that anyone can download, install and manage for free; but WebLab-Deployer enables users not willing to do the whole process to use instances of WebLab-Deusto deployed in remote servers. It created WebLab-Deusto[6] instances in external servers provided by the University of Deusto, so anybody could request an institutional deployment and use it. Internally, this was translated in running certain scripts that would create a whole WebLab-Deusto environment and database for the particular institution. As part of the Spanish mCloud project[7], where tools for migrating applications to the Cloud have been developed[8], this project has had a major upgrade to be compatible with certain Cloud technologies explained in the rest of the contribution, and it has been renamed as *wCloud*.

[5]http://wordpress.com
[6]http://weblab.deusto.es

IV. Cloud Computing

During the mCloud project, different Cloud technologies and approaches were analyzed. This section covers a summary of the decisions taken in the context of the mCloud project. The mCloud project is focused on making it possible to migrate regular applications to a Cloud environment [20], [21].

A. Public and private Clouds

A Cloud system heavily relies on virtualization technologies so a set of *real* servers virtualize a higher number of *virtual* machines. Using this virtualization, several advantages arise: a virtual machine can be disconnected from one real server and moved to other, its resources can be increased (e.g., adding or removing RAM, hard disk drive), it can be paused to make backups of the whole machine, etc. One of the most interesting features is the fact that you can resize the amount of resources dedicated to an application dynamically. For example, an application can be using a single server, and if the load of users is increased considerably affecting the quality of service, one or more virtual machines can be started and the load of users can be balanced among the different copies. As the load of users decreases, these virtual machines can be paused or stopped.

The most popular usage when refering to the Cloud is relying on companies providing virtual machines (in the case of IaaS –Infrastructure as a Service–) or application environments which after all are running in certain virtual machines (in the case of PaaS –Platform as a Service–), or services. This is considered a *public cloud*, since users are relying on companies which, at certain costs, are publicly providing access to their resources. However, it is also possible and useful to have *private clouds*, where a company can have a set of servers deployed indoors, and build a cloud on top of it.

For example, a company developing multiple applications can be interested on having its own infrastructure, and use a cloud technology to deploy the different applications on virtual machines that will be running and sharing the same resources. If there are peaks on one application, it could use a bigger amount of the resources during the peak, while other applications could take those resources in different moments. There are a set of technologies to implement this, but the most popular Open Source technology for this is OpenStack.

B. OpenStack

OpenStack[9] is the most popular technology for creating private and public clouds. It is developed by a big consortium of

[7]http://innovation.logica.com.es/web/mcloud
[8]https://github.com/mcloudpy
[9]http://www.openstack.org

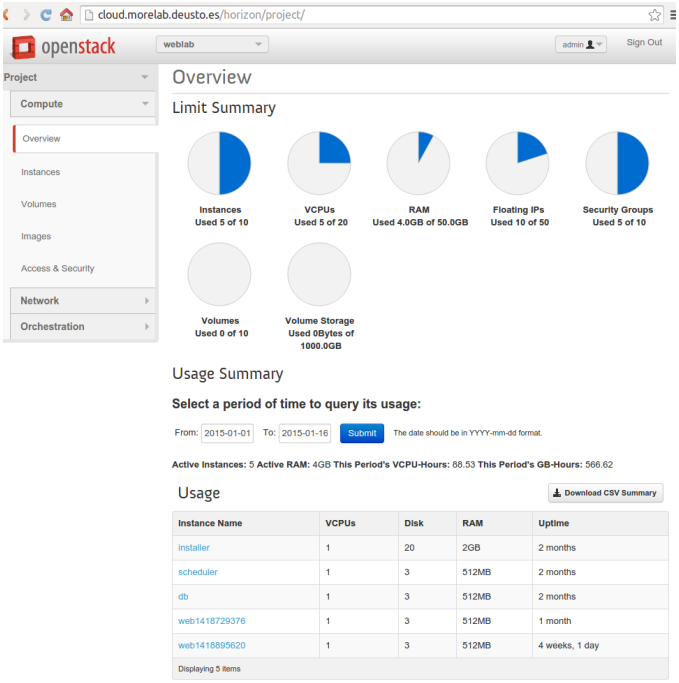

Fig. 3. OpenStack dashboard

companies including IBM, Intel, HP, Cisco Systems, Red Hat or Canonical among others[10], where the 8 platinum members must contribute each $500k USD plus 2 full time employees, in addition to the Gold members, corporate sponsors and others, becoming one of the largests Open Source projects.

In the context of mCloud, an OpenStack system has been deployed in three servers indoors, creating a private cloud. Through the user interface (see Figure 3), the administrator can create projects, asign resources to these projects, and let users managing those projects to create virtual hard disk drives, virtual networks, virtual machines, and connect each other. OpenStack provides also a REST programmable API that can be used to create applications on top of it, controlling the resources or implementing different policies. The system also provides advanced features in security or tools such as LBaaS (Load Balancers as a Service), which relies on HAProxy.

C. CloudFoundry

CloudFoundry[11] is an extensible, Open Source, Platform as a Service (PaaS) software which is becoming the most popular Open Source PaaS system. As opposed to OpenStack, this tool provides a development platform, where developers can upload their applications directly, developed in a number of programming environments. Typically, an administrator deploys CloudFoundry on top of a IaaS (which can be OpenStack, Amazon Web Services or others supported). Then, the administrator adds a set of common services, such as relational databases (such as MySQL, PostgreSQL), NoSQL databases (such as Redis, MongoDB) or queuing services (such as RabbitMQ). Once these services are registered, the administrator can add a set of programming frameworks available (e.g., Java servlets, Node.js or so). Finally, it will manage which users can deploy applications to which machines and access which services. Then, a developer can upload a Java application to the CloudFoundry instance, and internally the system manages the required virtual machines. Developers do not have access to the virtual machines running the programming platform, so they are working in other layer, where all the internals are managed by the IaaS, while some details can be tuned by the PaaS administrator depending on the selected IaaS.

While in the context of mCloud, CloudFoundry is used in other scenarios for its simplicity, it was considered that WebLab-Deusto would not suit in its current form the supported frameworks, so it was not used for the wCloud project.

D. Tools developed

So as to support some of the features missing in OpenStack (since it is a IaaS), some applications were implemented[12]. The most relevant ones are rstatus, which is an extensible load balancer system which relies on libcloud[13] (an Apache project to manage different IaaS environments such as OpenStack, RackSpace or Amazon Web Services using a single API), so it is possible to deploy a certain service on the virtual machines in OpenStack and in certain programmable circumstances (such as the load of users using WebLab-Deusto), load or stop virtual machines. Changes in libcloud were required and contributed to implement this tool.

V. wCloud

This section describes the wCloud system from a functional and technical perspective.

A. Functional description

The wCloud system[14] is an Open Source system[15] that creates WebLab-Deusto instances automatically as requested by users. To create a customized WebLab-Deusto environment, lecturers can log in and create an account. Once they have created an account, they will be able to provide an institution name, image, base URL, link and Google Analytics number, as well as some parameters such as the username and password of the administrator. Once provided, the creation process will be started, and a WebLab-Deusto instance will be available with such configuration. Figure 4 shows two screenshots of this process, and Figure 5 shows an example of customized WebLab-Deusto.

The following is a set of examples of the deployments created as part of the OLAREX project:

- https://cloud.weblab.deusto.es/w/nfsg/
- https://cloud.weblab.deusto.es/w/school32/
- https://cloud.weblab.deusto.es/w/abadinobhi/
- https://cloud.weblab.deusto.es/w/kmai/
- https://cloud.weblab.deusto.es/w/kharkov/

[10] http://www.openstack.org/foundation/companies/
[11] http://www.cloudfoundry.org
[12] https://github.com/mcloudpy/
[13] https://libcloud.apache.org
[14] https://cloud.weblab.deusto.es
[15] https://github.com/weblabdeusto/weblabdeusto/

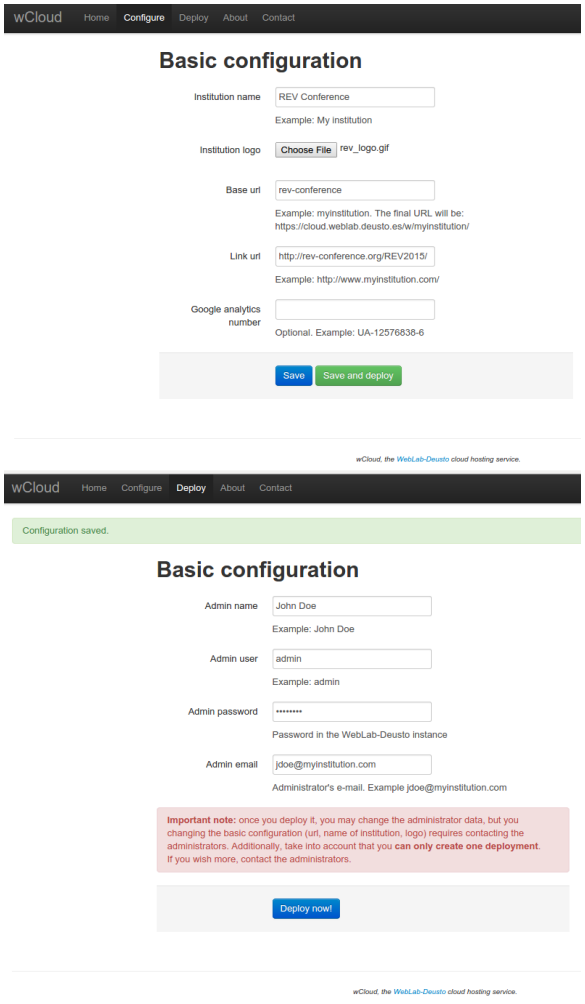

Fig. 4. wCloud user interface when creating a new instance

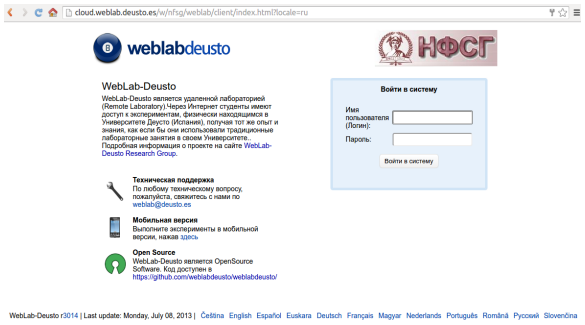

Fig. 5. Customized WebLab-Deusto system for NFSG

B. Technical description

A typical deployment of WebLab-Deusto uses a MySQL database[16] for persistent storage (users, permissions, analytics) and a Redis database[17] for scheduling. Redis is an efficient NoSQL database that stores information in memory and provides a small set of data types with atomic operations. So as to support user load balancing, WebLab-Deusto runs in multiple processes, and an Apache server proxies the requests to the different processes. Since sessions are temporally stored in memory in each process, Apache uses a technique called *sticky sessions* to manage it. In this technique, each process stamps on the first request a cookie establishing that the requests of this particular session are managed by that particular process. This way, clients will send it in the following requests, and Apache, if the cookie is present, will always forward the requests to the selected process. More details are available in the documentation[18].

In its origins, WebLab-Deployer was a Flask application that relied on:

- A pool of available databases. There was a customizable number of available databases (e.g., 1000 databases created called wcloud0000..wcloud1000). wCloud in its own database had an internal counter and whenever somebody created a new WebLab-Deusto instance, the counter would be increased and that instance would use one of those databases.
- A local server run as root (administrator user in UNIX systems) that was waiting for HTTP requests from the same server. Whenever a new WebLab-Deusto system was created, it would add a file with certain Apache configuration and would reload the Apache server configuration.
- A local server that managed a memory queue and would be creating the deployments. Basically, if multiple users create a WebLab-Deusto environment, they are queued in this process.

The system was not reliable and there were issues between the different components, so it was redesigned. wCloud is nowadays a Flask application, but Celery[19] (a distributed task queue which can be deployed on top of RabbitMQ[20] or Redis) is used to avoid managing the queue directly. This way, there is no need to have two more servers: one process as root is waiting for Celery tasks in the queue that establish that the Apache configuration must be reloaded, and it reloads the Apache configuration; other process is waiting for Celery tasks that require. Additionally, databases can be created automatically as long as they have a prefix, and this way keeping such database pool is not required anymore.

In addition to this, wCloud was tested in the OpenStack in Section IV-B. As seen on Figure 6, five Virtual Machines were started with WebLab-Deusto deployed. One of the virtual machines contained the shared databases (both Redis and MySQL), while in the future Trove[21], the OpenStack DBaaS (Database as a Service) service should be used, while a new version would be required. The rest support the different components of WebLab-Deusto and wCloud, being able to create more web instances in more virtual machines.

C. Required changes in WebLab-Deusto

Already for the original WebLab-Deployer system, it was required to support that a WebLab-Deusto system, including

[16] http://www.mysql.com
[17] http://redis.io
[18] http://weblabdeusto.readthedocs.org
[19] http://www.celeryproject.org
[20] http://www.rabbitmq.com
[21] http://docs.openstack.org/developer/trove/

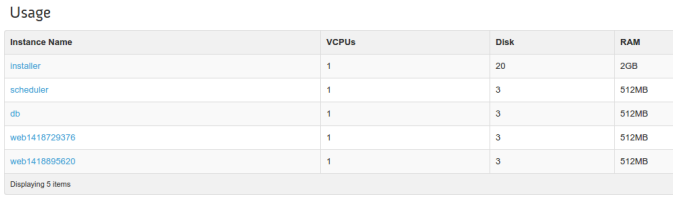

Fig. 6. Five Virtual Machines deploying WebLab-Deusto

its database, etc. could be created in a fully automated process, with no human interaction. However, so as to fully embrace the cloud environment paradigm, it is mandatory to support other features, of which most have been implemented:

- Reduce the number of ports used. WebLab-Deusto used to create 10 ports per process for different tasks (supporting JSON web services, SOAP, XML-RPC, web, etc. in different ports and then rely on the proxy server so as to only use the 80 or 443 port). Given that the number of ports is limited in a server, we have reduced this number to 3 at the time of this writing. Ongoing efforts to have a single one per process or even only one per deployment by relying on WSGI servers such as gunicorn[22].
- Move parameters that were stored in configuration files to the database. In the past, all the experiment configuration was stored in a set of JavaScript files that, if a university is hosting the WebLab-Deusto system, could easily change. However, through wCloud, users can not manipulate files directly, so these settings were moved to the database and added to the administration panel.
- Migrate the client technology. The WebLab-Deusto client, at the time of this writing, is developed in Google Web Toolkit (GWT)[23]. Using this technology, upgrading any change on the client requires a 10 minutes compilation to generate all the required JavaScript and HTML files in all the languages and web browsers. While this is not a relevant issue when deploying it in a local server for a single WebLab-Deusto deployment, it becomes a problem when automating the process of deployment of WebLab-Deusto in virtual machines. A new WebLab-Deusto client has been developed but it is still under testing at the time of this writing.
- Store scheduling information in the database. WebLab-Deusto used to require administrators to edit a couple of Python files to map which laboratories were using which external systems. However, in wCloud, where administrators do not have access to these files, the number of remote laboratories is unfortunately fixed. To avoid this problem, WebLab-Deusto now stores this information in the database and provides a administration panel where administrators can, through the web interface, select which external laboratories can be used.

[22]http://gunicorn.org/

[23]http://www.gwtproject.org/

VI. Conclusions

This contribution describes the wCloud system, which is a RLMS as a Service software system that enables schools and universities to create customized remote laboratory management systems automatically. With them, users can create a customized WebLab-Deusto system and access laboratories in the University of Deusto or other universities providing remote laboratories through WebLab-Deusto. The contribution shows how this system, developed as part of the mCloud project, has been used to create WebLab-Deusto instances in multiple schools and universities.

It also describes what are the technical decisions behind this software system and the characteristics of the private cloud environment on which it is going to be deployed.

Regarding future work, while wCloud and WebLab-Deusto have been adapted to a particular private cloud environment, it is not yet used in production in this environment since the private cloud deployment is used for testing purposes. Its adaptation to other public cloud environments such as Amazon Web Services will be considered so other users deploying WebLab-Deusto can efficiently deploy it in such schemes.

Acknowledgment

This work has been supported by research grants IPT-2011-1558-430000 (mCloud) funded by the Spanish Ministerio de Ciencia e Innovación.

References

[1] O. Dziabenko, J. García-Zubia, and I. Angulo, "Time to play with a microcontroller managed mobile bot," in *Global Engineering Education Conference (EDUCON), 2012 IEEE*. IEEE, 2012, pp. 1–5.

[2] L. Gomes and S. Bogosyan, "Current trends in remote laboratories," *Industrial Electronics, IEEE Transactions on*, vol. 56, no. 12, pp. 4744–4756, 2009.

[3] C. Gravier, J. Fayolle, B. Bayard, M. Ates, and J. Lardon, "State of the art about remote laboratories paradigms-foundations of ongoing mutations," *iJOE*, vol. 4, no. 1, 2008.

[4] B. Carisa, A. Burain, S. Molly H, and C. Lawrence, "Running control engineering experiments over the internet," 1995.

[5] B. Aktan, C. Bohus, L. Crowl, and M. Shor, "Distance learning applied to control engineering laboratories," *Education, IEEE Transactions on*, vol. 39, no. 3, pp. 320–326, 1996.

[6] J. Henry, "Running laboratory experiments via the world wide web," in *ASEE Annual Conference*, 1996.

[7] A. Coble, A. Smallbone, A. Bhave, R. Watson, A. Braumann, and M. Kraft, "Delivering authentic experiences for engineering students and professionals through e-labs," in *Education Engineering (EDUCON), 2010 IEEE*. IEEE, 2010, pp. 1085–1090.

[8] R. Cedazo, F. Sanchez, J. Sebastian, A. Martínez, A. Pinazo, B. Barros, and T. Read, "Ciclope chemical: a remote laboratory to control a spectrograph," *Advances in Control Education–ACE*, vol. 6, 2006.

[9] J. Del Alamo, L. Brooks, C. McLean, J. Hardison, G. Mishuris, V. Chang, and L. Hui, "The mit microelectronics weblab: A web-enabled remote laboratory for microelectronic device characterization," in *World Congress on Networked Learning in a Global Environment, Berlin (Germany)*, 2002.

[10] D. Gillet, H. Latchman, C. Salzmann, and O. Crisalle, "Hands-on laboratory experiments in flexible and distance learning," *Journal of Engineering Education*, vol. 90, no. 2, pp. 187–191, 2001.

[11] I. Gustavsson, J. Zackrisson, L. Håkansson, I. Claesson, and T. Lagö, "The visir project–an open source software initiative for distributed online laboratories," in *Proceedings of the REV 2007 Conference, Porto, Portugal*, 2007.

[12] Z. Nedic, J. Machotka, and A. Nafalski, "Remote laboratory netlab for effective interaction with real equipment over the internet," in *Human System Interactions, 2008 Conference on*. IEEE, 2008, pp. 846–851.

[13] R. Safaric, M. Truntič, D. Hercog, and G. Pačnik, "Control and robotics remote laboratory for engineering education," *International Journal of Online Engineering (iJOE)*, vol. 1, no. 1, 2005.

[14] F. Torres, F. Candelas, S. Puente, J. Pomares, P. Gil, and F. Ortiz, "Experiences with virtual environment and remote laboratory for teaching and learning robotics at the university of alicante," *International Journal of Engineering Education*, vol. 22, no. 4, pp. 766–776, 2006.

[15] J. Hardison, K. DeLong, P. Bailey, and V. Harward, "Deploying interactive remote labs using the ilab shared architecture," in *Frontiers in Education Conference, 2008. FIE 2008. 38th Annual*. IEEE, 2008, pp. S2A–1.

[16] P. Orduña, P. Bailey, K. DeLong, D. López-de Ipiña, and J. García-Zubia, "Towards federated interoperable bridges for sharing educational remote laboratories," *Computers in Human Behavior*, vol. 30, pp. 389–395, Jan. 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0747563213001416

[17] P. Orduña, "Transitive and scalable federation model for remote laboratories," Ph.D. dissertation, Universidad de Deusto, Bilbao, Spain, May 2013. [Online]. Available: http://paginaspersonales.deusto.es/porduna/phd/

[18] P. Orduna, A. Almeida, D. Lopez-De-Ipia, and J. Garcia-Zubia, "Learning analytics on federated remote laboratories: tips and techniques," Apr. 2014, 00000.

[19] P. Orduna, X. Larrakoetxea, D. Bujan Carballal, I. Angulo, O. Dziabenko, L. Rodriguez-Gil, D. Lopez de Ipina, and J. Garcia-Zubia, "WebLab-deployer: Exporting remote laboratories as SaaS through federation protocols," Sydney, Australia, Feb. 2013, pp. 1–5. [Online]. Available: http://www.weblab.deusto.es/web/images/publications/rev2013_wcloud.pdf

[20] L. Orue-Echevarria, J. Alonso, M. Escalante, and S. Schuster, "Assessing the readiness to move into the cloud," in *Cloud Computing*. Springer, 2013, pp. 12–20.

[21] L. Orue-Echevarria, M. Escalante, and J. Alonso, "An assessment tool to prepare the leap to the cloud," in *Cloud Computing*. Springer, 2013, pp. 273–292.