Putting Reusability First: A Paradigm Switch in Remote Laboratories Engineering

doi:10.3991/ijoe.v5i1.816

Christophe Gravier, Jacques Fayolle, Jérémy Lardon,Bernard Bayard, Gaétan Dusser, Romain Vérot
TELECOM Saint-Etienne, University of Saint-Etienne, University of Lyon, France

***Abstract*—In this paper, we present a new devices brought online thanks to our Collaborative Remote Laboratories framework. Whereas previous devices integrated in our remote laboratory belongs to the domain of electronics, such as Vector Network Analyzers, the devices at the concern in this paper are, on one hand, an antenna workbench, and on the other, an homemade switching device, which embeds several electronic components. Because the middleware and framework for our environment were designed to be reusable, we wanted to put it to the test by integrating new and different devices in our Online Engineering catalog. After presenting the devices to be put online, we will expose the software development efforts required in regards to the reusability of the solution. As a consequence, the expose work and results tend to make the Online Engineering software architects to think reusability first, breaking with the current trends to implement Remote Labs one after the other, without much reusability, apart the capitalized experience. In this, we defend a paradigm switch in our current engineering approaches for Remote Laboratories implementations: Reusability should be thought first.**

***Index Terms*—Computer Supported Collaborative Work, Distance Learning, Online Engineering, Remote Laboratories.**

I. INTRODUCTION

In this paper, we are primarily interested in the software reusability for (collaborative) remote laboratories. We expose how we can achieve a certain degree of reusability for software solution, which allows remote hands-on approaches for online universities. This consideration is compulsory for the leverage of online laboratories (would they be remote or virtual), given the high development cost associated to such solutions. This field of interest focuses on the missing, but emerging, link in the widespread online education schema (practical works that are usually postponed to in situ sessions).

The major issue is to cope with the lack of reusability of the underlying software. The most common approach is to build proprietary software for a given online laboratory, and then, when the opportunity arises, to see how it could be adapted for a new device to put online. The second under explored way of thinking is to think and create in a reusable way from the very former developments, thus creating a reusable framework for the given education task. This paper addresses the latter one.

The goal of this paper is to try to identify, on the basis of our experience, what we think as reusable links and software in the software chain of remote laboratories. On our observations, and the resulting framework previously exposed, we try to put our architecture to the test by applying reusability to a brand new different collaborative remote laboratory. The use case throughout the paper is an antenna workbench used in distance learning.

The paper is organized as follows. Section 2 stresses historical aspects of our research that lead us to a reusable framework. Section 3 dresses the picture of the custom Antenna workbench developed at our University, which will be the device involved in the CRL exposed here. Section 4 brings key ideas on the software approach used and how it performs in term of reusability. Section 5 concludes.

II. HISTORICAL FOUNDATIONS OF THIS PROJECT

We started a project in 1999 for the remote control of some of our research laboratories devices [1]. At first, this work was lead by the need to share devices between different research laboratories and engineering schools, which all have participated in their funding. Instead of moving the device itself to every place where it is required, or bringing the researchers to the device in our university each time this research requires an access to the device, we decided to build a complete software solution so that every participant in the consortium can gain access to the device easily, by providing the lower financial effort possible.

As a consequence, we built a complete dedicated software solution that has been first used in 2001, where the involved device was a Vector Network Analyzer from Hewlett-Packard series.

This solution consisted in proprietary software developed in our laboratory for this very specific need. The communication relied on a custom application protocol, while distributed computing and middleware paradigm applied to distributed applications were at the earliest stages. Like other former solutions at that time, the communication was point-to-point, over a TCP/IP dedicated application link, following direct socket programming widespread at that time.

This approach was therefore developed in the context of our collaboratories. We rely on the definition of collaboratories from Cerf, in 1993: « [a] center without walls, in which the nation's researchers can perform their research without regard to geographical location, interacting with colleagues, accessing instrumentation, sharing data and computational resources, and accessing information in digital libraries. » [2].

Since the platform was fully operational for research purpose, the idea of using it for distance education emerged from the leverage of distance learning campus at the French national level.

Actually, France efforts were made since early 2001 in order to promote online campus. The associated national funding aimed at developing partially or fully distance teaching plans in French Universities.

As a consequence, we integrated in online campus environment the existing remote experiment platform, based on the same Hewlett Packard Vector network Analyzer (henceforth VNA), thanks to minor software adjustments. Such a piece of software allowed us to conduct remote hands-on approaches, without the need for student's to come back to our engineering school for in situ learning sessions [3].

Nonetheless, when there are bugs in a software solution, users ask for corrections. Once the corrections are made and the system is running smoothly, the users usually ask for more advanced functionalities. As a consequence we were asked to bring other devices online.

We tried to reproduce what had been done for the Hewlett Packard VNA, for other devices such as another VNA of Anritsu brand and an Antenna workbench (a home made devices with two antennas, which can be translated around two axis).

At that time, on the basis of our Remote Laboratories experience and software development of Remote Laboratories platform, we realized at our expense what we consider as two major stakes for Remote Laboratories [4] :

- Lack of collaboration among students, unlike local laboratories,
- Low reusability of the software solutions.

These two observations lead us to search deeply in the literature. However, we could hardly find a remote laboratories platform embedding synchronous collaboration among students or reusability of the software architecture using a high level of formalization for fewer future software developments.

At that time, we had a single widespread point of view in the online engineering community: building new Remote Laboratories from scratch, and to seek software reusability when possible.

At the opposite, we have chosen in 2004 to propose a new approach: to build new Remote Laboratories from a common framework (not related to any device), and to seek dedicated software developments when reusability was not possible.

This difference of thinking tends to maximize the reusability of the framework since the software architect is thinking at what can be reused at first, and only then he gathered missing software pieces, instead of creating software from scratch and thinking of new functionalities we can import from previous development. Another important aspect is that there is one single framework, which means that a correction is made in a single software, instead of having to reengineered each piece of software involved in every remote laboratories previously put into production.

Those studies, which also heavily explored the collaboration aspects between students [5], which won't be debated in this paper, lead to the creation of a brand new original framework, allowing fast development of Collaborative Remote Laboratories (henceforth CRL) [6].

We now have reached the state of a functional and operational CRL framework, hopefully reusable. We are currently using the framework with a Vector Network Analyzer[1]. We think our framework is reusable enough to support CRL from a complete different domain. That is the reason why we want to put the framework to the test by trying to implement a CRL involving an Antenna workbench, using the already created framework. This paper will therefore report how to cope with the heavy software development efforts to make a brand new CRL online.

III. ANTENNA WORKBENCH

This section deals with the antenna workbench that has been set up in our laboratories in 2002, for distance learning purpose.

The mechanical aspects involved in the antenna workbench are divided into three main parts: a mobile antenna, a rotary antenna and a horizontal carriage upon which a vertical carriage is fixed. In order to get an idea of the size of the workbench, we provide here some figure on the workbench. The average distance between two antennas is 160 cm (≈63 inches). The two end stops sensors are separated by a distance of 200 cm (≈78,75 inches). The complete width of the antenna workbench is 250 cm (≈98,43 inches). Every element's movement is controlled by steppers, which are controlled by ELSTEP power modules. Figure 1 provides a complete view of the antenna workbench with the three different parts that composed the workbench.

The mobile antenna of the workbench is acting as a receptor. Its movement is only horizontal. For safety and security reasons, it is important to enforce a software validation for the movement of this antenna. The movement of the associated antenna must not overrun 112 cm (≈44 inches). The position of this antenna is monitored by several sensors installed all along the workbench's rail (c.f. Figure 2).

The rotary antenna of the workbench is acting as a transmitter (Figure 3). Like the mobile antenna, a stepper controls the rotary antenna. It allows rotation varying from $-2\pi/3$ up to $+2\pi/3$ angle. Some sensors located above the workbench's rail are used to detect when the antenna reaches the 0 degree angle and the maximum angles. This antenna is fixed above the workbench's trays, using a plastic arm (metal has been avoided because of electro-magnetic perturbations).

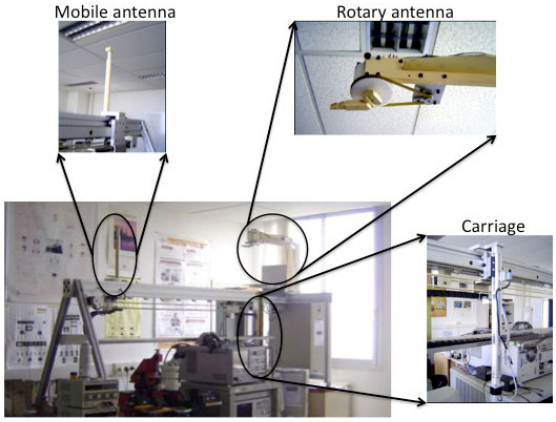

Figure 1. The antenna workbench and position of the three different modules.

[1] http://diom.istase.fr/satin/einst/einst_demo.avi

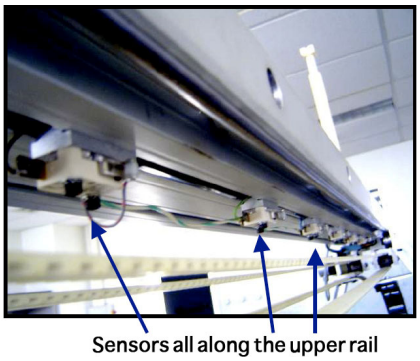

Figure 2. Sensors for the detection of the mobile antenna location.

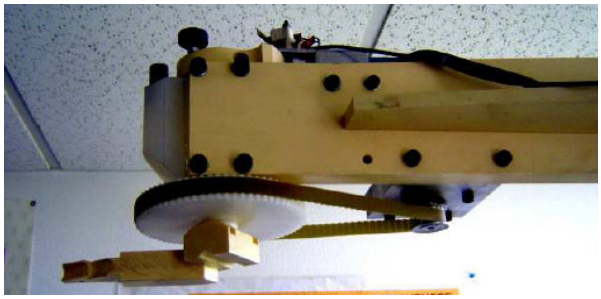

Figure 3. Close caption of the rotary antenna (transmitter).

Finally, the last element is the carriage, i.e. the obstacle elevator. It will be put between the mobile antenna and the rotary antenna, and that will modify the received signal, and therefore the measurement conducted by the students, in order to learn how an obstacle interferes with the radio transmission. The learners should be able to move the mobile antenna as well as the obstacle and draw the transmission rate depending on those elements location. This elevator relies on 10 sensors (on the horizontal axis), which will transmit the location of the tray during the loading of an obstacle on the antenna workbench's rail. A complete figure of the antenna workbench is drawn in Figure 3. This figure illustrates the position of the different sensors, which are the only feedback that the software can acquire from this specific device.

On the basis of this description, we will now expose the procedure to follow in order to integrate a new Collaborative Remote laboratory in our framework at the lower software development cost.

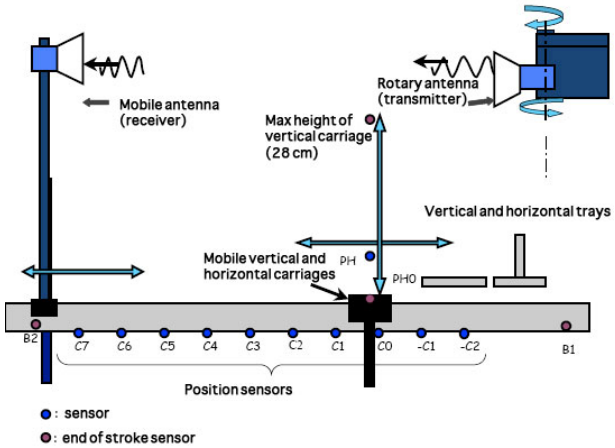

Figure 4. Sketch of the antenna workbench and its different elements and sensors.

IV. INTEGRATION IN THE CRL PLATFORM

In section 2, we exposed our motivations for building a reusable collaborative remote laboratories platform. Section 3 brought details on a new device to put online using our platform. The current section will explain the steps for the integration of the antenna workbench in our CRL platform. We will try to illustrate what succeeded in factorizing at the software development level, and what is harder to reuse from one collaborative remote laboratory to another.

When observing traditional Remote Laboratories configurations, there are usually the same four elements involved in the software chain: the device to remote control, the local computer, the remote computer, and the user himself. On this observation, three different links in the software chain can be identified:

- Device to local laboratory computer link: "device-to-computer interface".
- Local laboratory computer to remote computer: the Middleware,
- Remote laboratory to the Users link: the Human-Computer Interface.

Those four elements and the associated three interfaces are illustrated in Fig. 4.

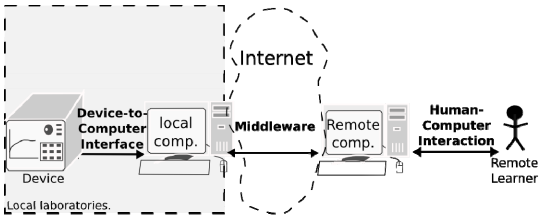

Figure 5. Software chain of traditional Remote Laboratories.

We will now discuss what is being reused in our framework, in other words what does not require reengineering for every new Remote Laboratories. This study will cover each previously identified link of the Remote Laboratory software chain.

A. *Human-Computer Interaction (HCI)*

What are the expected functionalities from Human-Computer Interaction for Remote Laboratories? During the last decade of our experiences in the domain, we tend to answer the following list:

- Accuracy and faithfulness to the original HCI of the real device it corresponds,
- Middleware connectivity for sending command et retrieving results,
- Fullness of the GUI widgets allowed by the framework,
- Assisted construction of the GUI (WYSIWYG if possible)
- Widgets independent to device (that is the assembling of the different widgets that creates the remote GUI, but each widget itself is not dedicated to any device in order to favor reusability).

In order to address all of those objectives, we chose to handle the HCI module in our reusable framework using Semantic Web, and especially Ontologies. In order to motivate the use of Ontologies for the HCI description, we

must expose a current trend for Remote Laboratories. This is important in order to understand our motivations in the HCI module construction. Remote Laboratories are beginning to interoperate with Learning Management Systems (LMS) [7]. LMS hold pedagogical scenarii of online exercises. Those scenarii can be related to a lecture, online questionnaires or even online practical work. Nonetheless, unlike lectures or online exercises, remote practical work needs a strong interaction between the system, which runs the scenario, and the environment of the online practical work itself. The main reason lies in the learning scenario, which has to retrieve sequences of actions performed by the users, in order to infer what the learners are trying to do. When LMS correctly infer the learner's intentions, it can therefore provide an appropriate help to the connected users, or even personalized the running scenario accordingly. Thus enhancing the learning process. The need of inference implies that a strict descriptive approach[2], which may look simpler at first sight (XML serialization for example), is not enough to interoperate remote laboratories and LMS. On the contrary, when the HCI benefits from a logic representation, it is possible to provide a specialization of Remote Laboratories GUI concepts to LMS for inference.

Because we are forward thinking to link LMS with Remote Laboratories platforms in future work, we therefore need a semantic approach at the HCI in level, in order to further sustain the associated Knowledge Base.

As a consequence, we have created an ontology of graphic elements used in Remote Laboratories HCI, and serialized it in an OWL[3] file. This file is accessible online upon browsing http://dev.istase.fr/satin/rlab/eINSTv7.owl

We are then able to create a JAVA program that parses the OWL file and create the associated HCI. It is important to note that the JAVA program is independent to the device remotely controlled. As a consequence, there is only the ontology itself, which has to be created for the creation of the HCI of one new Remote Laboratory in our framework (XML serialization). No line of code is required for this step.

Nevertheless, even if no line of code is required, the edition of OWL files is not so simple, especially for a Remote Laboratory teacher, not necessarily specialized in Computer Science. There are three different possibilities for the edition of the OWL file. The first one consists in writing the plain OWL file by using a text or XML editor. This is neither an elegant nor an easy way of doing this, due to the verbosity of OWL format. Errors can occur from copy/paste or typography errors can make it unworkable. The second option is a common program used for writing ontology, which is called Protégé[4] and maintained by Stanford University. While Protégé is a good solution for editing top-level ontologies, we think that domain ontologies, bound to a dedicated specialization of concepts, benefit from the use of dedicated software. For example, it sounds easier to edit the ontology of the GUI with a WYSIWYG editor. This is the reason why we developed such a WYSISWYG editor in our research project. This GUI qualifier tool parses the device ontology available online and let the user drag and drop widgets, which are individuals in the ontology, from the widget toolbar to the visual panel. The resulting GUI is serialized in an OWL file, ready to be operated by our JAVA program.

This WYSIWYG editor eases the development of the client application for Remote Laboratories since no additional line of code is required for any new device to put online in a Remote laboratory context. The Remote Laboratory architect has only to provide the OWL file resulting from the WYSIWYG editor in order to enjoy a ready-to-go GUI.

B. Middleware

The second link in the traditional software chain of Remote Laboratories is between the remote computer in front of the user, and the local computer connected to the remote device. This is the middleware upon which commands, requests and answers are transported from the users to the local laboratories and vice versa. The main quality of a middleware is transparency. The more the middleware is transparent for the user and the software architect, the best it is assumed to be. In the field of Remote Laboratories, we expect the middleware to be transparent, but also as lightweight as possible in term of bandwidth, at least at the client-side. Actually, from our experience, Remote Laboratories involve learners in their student room where they may not have sufficient bandwidth, or university in foreign countries with limited bandwidth. We have therefore thought about current middleware paradigm such as RPC[5], OOM[6] and MOM[7]. Because MOM do not expose methods to remote object but on an applicative mailing paradigm, they therefore ensure:

- Device safety (protection of remote object if access is through messages exchanges and not object proxy manipulation as when using OOM),
- Lightweight data exchange since there is only messages that are exchanged and not a serialization of data objects in memory between the clients and the server.

In our framework, the MOM is coupled with an Applications Server (AS). This association guarantees:

- Ordered message delivery,
- Asynchronous call when using Publish/Subscribe paradigm,
- Robustness because of transaction mechanisms provided by the AS,
- Protection of the remote device (the remote device is not directly accessed from the outside network, but only through the AS authorization and authorization service thanks to a LDAP directory).

The entire framework is based on J2EE technologies.

The key idea for this link is that our widgets depicted in the online ontologies (c.f. 3.1) are embedding the connection to the middleware. That is to say that they inhibit the application listeners and fire events on the

[2] In the meaning of Tim Berners-Lee semantic web stack [8], Even if we can wonder if this is really a stack [9].

[3] OWL : Web Ontology Language, http://www.w3.org/TR/owl-features/

[4] http://protege.stanford.edu

[5] Remote Procedure Call

[6] Object Oriented Middleware

[7] Message Oriented Middleware

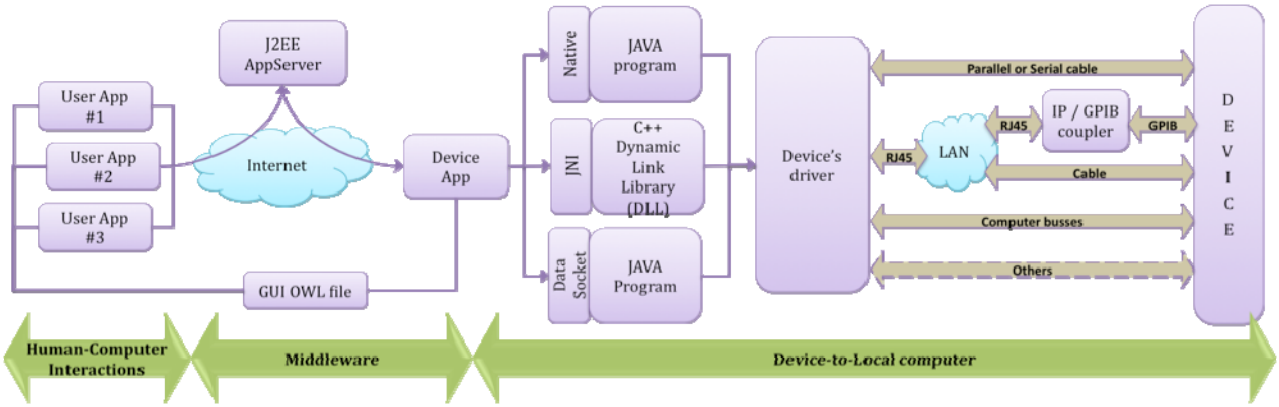

Figure 6. Three different connectors and their associated place in the framework (Detailed view from links detailed in Fig. 5)

server-side instead. As a consequence, each time one of our widgets is chosen in the HCI ontology, it is automatically associated at runtime with the middleware. That is to say that our widgets have natively data exchange capacities over the middleware. In other word, the software architect does not have to provide any additional line of code in the matter of middleware; it is already entirely embedded in the native CRL framework widgets.

C. *Device to local computer interactions*

Establishing a reusable link between the local computer and the device itself required to master standards and norms for the software access to the instruments. This is hardly possible as the drivers implementing the connection to the device are heterogeneous, and due to the exponential growth of the number of involved technologies. This also depends on the physical interfaces to employ: PXI[8], USB[9], GPIB[10], …

Nevertheless, some devices, of different constructors or series, offer the same functionalities, it is therefore harmful to reusability to propose a single and dedicated interface library per device in our framework. It could be more relevant to partition the set of targeted devices as classes, on the criteria of their physical and software interfaces, in order to save software developments for the same functionalities, but different physical devices.

This is the main idea that leads to the development of new standard, such as VISA[11], IVI[12] and LXI[13] [10]. These technologies aim to make devices drivers as reusable as possible, that is to say as independent (to the device they provide an interface to) as possible. The purpose was to group devices in different categories.

Unfortunately, those technologies suffer from several drawbacks that prevent to generalize their use for every situation. First, those are young standards towards computer-assisted instrumentation. That is to say that they require devices compliant to the new standards, which obviously exclude legacy system and "old" device. Age in instrumentation should be considered to be several years given the depreciation implied by the cost of high technological instruments. Although past attempts to build a bridge between legacy systems and modern standard has been made [11], the gateway is not perfect and few research centers would drop his park of machine in favor of a recent one only for proving a mean for new software developments. Other limitations lead us to show prudence in reusability abilities of the link between local computer and the device itself. Landragin & Schwartz ([12]) expose three of them:

- The results will not be the same between devices whose measurement performances are different whereas they seems perfectly identical,
- Modules with specific and unique functions in their categories will not be usable using IVI drivers, as IVI does not allow interchangeability in the context of specific functions,
- The IVI layer introduced is likely to slow the system towards low-level calls, which may be an issue for real-time environments.

We can also add another limitation we have encounter in our experiences: several remote laboratories involved a homemade device, which may not be compliant to standards such as IVI. This is an issue that applies to the antenna workbench exposed in section 3. As a consequence, the software link between the local computer and the device itself is hardly reusable, and it mostly depends on the device remotely control.

Another problem is the difference of technologies to use in order to connect the Remote Laboratories platform to the remote device.

There is no denying that all existing and remote controllable devices do not share the same technologies of communication: some requires direct and specific programming (C++, Datasocket, etc.), and other, because they are compliant to some emerging norms, can be address in the programming language implementing a standards, which still require some programming.

In order to cope with the issue of the many and heterogeneous amount of technologies allowing to connect a device to a middleware, we decided to implement different connector, which enables to plug different technologies to our Collaborative Remote Laboratories platform. The middleware is able to rely commands to devices on LAN, Serial port, Parallel port, etc. without the need to write additional code. This is possible thanks to the deployment of connectors written in

[8] PCI eXtensions for Instrumentation
[9] Universal Serial Bus
[10] General Purpose Interface Bus
[11] Virtual Instrument Software Architecture
[12] Interchangeable Virtual Instruments
[13] LAN extension for Instrumentation

Java and C++, which can trigger communication with devices who can discuss with Java, C++ or Datasocket (Figure 6). This implies openness of the platform in term of technologies, but by no mean heavy reusability of this link: specific developments are still compulsory in order to adapt the communication to specific devices. This is a strong problem to overcome in the future for this platform to be fully loosely coupled with existing devices.

As for the antenna workbench, whereas the software development cost was insignificant for the HCI and the middleware, the link between the local computer and the device is measured in months and number of line of code (cf. figure 7).

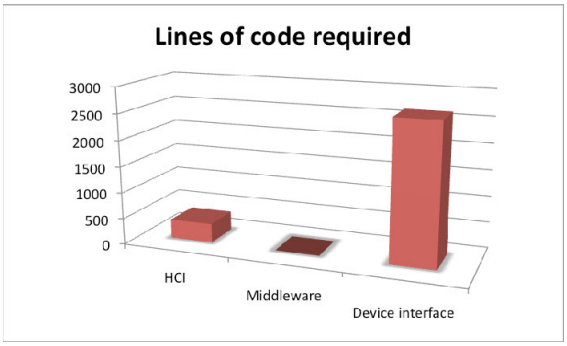

Figure 7. Software developments (number of line of code[14]) of the antenna workbench remote laboratory integrated in the reusable framework

V. ANOTHER EXAMPLE: HOME-MADE DEVICE BROUGHT ONLINE

We created a device to be coupled with a Vector Network Analyzer. The problem during remote hands-on approaches with this particular device was to cope with local manipulation on the device, e.g. to switch from one component being characterized to another. At the earlier steps of the project, a technician was required in front of the device, in order to perform the necessary operation of switching from one studied component to another, depending on the learners' walkthrough in the learning scenario. For each question, the technician had to replace the current component with the new one to be studied. In order to overcome this limitation (he presence was compulsory throughout the entire remote collaborative learning session, albeit the very few number of solicitations), we integrated all the components on a board, with the possibility to switch from one circuit to another using proper inputs (Serial cable). Figure 8 is a picture of this homemade device. The device was therefore called "WebSwitch".

The board has two switches (coaxial) of 6 input-output each. The 6 different associated circuits which can be switched to are the following:

- open-circuit
- short circuit
- a resistor
- a capacitor with a resistor
- an inductance
- an Operational Amplifier.

[14] Number of line of code includes comments. Lines of code for HCI are issued from the WYSIWYG editor, not plain text edited.

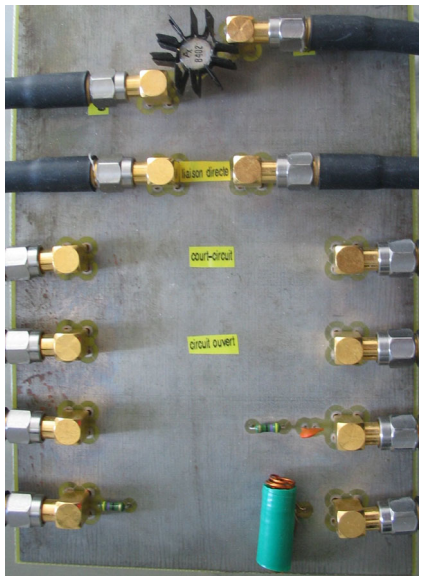

Figure 8. Picture of the "WebSwitch", a home-made device for switching from one component to be characterized to another.

We wanted to make this device available online, so that the remote tutor could be able to switch himself from one component to another, without breaking the learning flow, and without asking and waiting for an external operation lead by a local person. As a consequence, we tested the reusability of the platform by integrating the so-called "WebSwitch" into the platform, without much effort (the device-to-local computer link here didn't required a lot of software development as the device only propose a limited set of functionalities). This have been possible because are connector the devices in the platform are not bound to a specific technology (PXI, GPIB, etc.), but instead opened to any device and technology which can be employed through Java or C++ programming.

The resulting Graphic User Interface is presented in Figure 9, and has been chosen to be very close to the one of the Vector Network Analyzer, which are both usually coupled in the Remote Laboratory learning session.

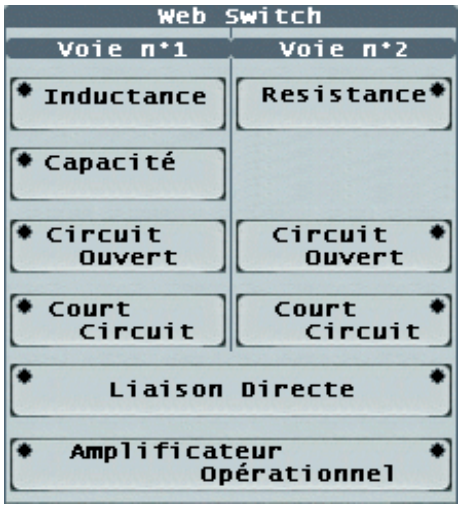

Figure 9. Capture of the GUI of the WebSwitch on the users' computer

VI. CONCLUSION

In this paper, we presented a previously developed framework for Collaborative Remote Laboratories (CRL). Based on the description of an antenna workbench and a homemade device, both used in remote laboratories, we tried to put our framework to the test by trying to measure the degree of reusability we have reached so far (how fast can a new device, for which the platform was not especially designed for, be integrated). On the observation of the widely accepted distributed architecture for traditional remote laboratories, we have measured for each link (Human-Computer Interactions, Middleware, and device interface), the effort that has been made to create the remote laboratory.

The main conclusions are about the time and the software development cost associated to those links. Whereas we achieved good software reusability at the HCI and middleware levels, the link between the local computer and the device itself is hardly reusable. This issue can be partially coped by using IVI drivers, for example, it is however hardly possible when the architect faces home made workbenches, which may not be compliant to such standards, or real-time systems when very latency is compulsory (IVI abstraction layer is time consuming at the execution time).

Our main goal is to decrease the integration of a new device as a Collaborative Remote Laboratory using the framework down to less than 2 days. So far, HCI and middleware connectivity require one day (write an instance of our GUI ontology for the HCI, and nothing to write for middleware as the widget embeds connectivity to our platform). This is already a significant advance for online engineering integration. Nevertheless, the device-to local computer link still requires several months for complex devices. Future work will consist in finding solution to cope with this last link in order to achieve a higher level of reusability.

REFERENCES

[1] B. Bayard, J. Fayolle, B. Sauviac and G. Noyel, “Projet Web Analyzer : Instrumentation réelle sur le world web”, CETSIS EEA, Clermont Ferrand, France, 29-30 octobre, 2001

[2] Cerf V. G., National Collaboratories: Applying Information Technology for Scientific Research. Washinton, DC: National Academy Press, 1993.

[3] Fayolle, J., *Evaluation of the different costs of the virtual universities*, Information Technologies in Higher Education, vol. 1, N° 2, pp. 56, 2004

[4] Gravier, C., Fayolle, J., Lardon, J., Noyel, G. *A Distributed Online Laboratory System for Distant Learning*, Proceedings of IEEE/ACM The International Conference on Signal-Image Technology and Internet-based systems (SITIS'2006), Hammamet, Tunisia, December 17-21, 2006.

[5] Gravier, C., Fayolle, J., Bayard, B. *Coping with collaborative and competitive episodes within collaborative remote laboratories*, Proceedings of Remote Engineering and Virtual instrumentation (REV'08), Duesseldorf , Germany , June 22-26, 2008

[6] Gravier, C., Fayolle, J., Bayard, B., Ates, M., Lardon, J., *State of the Art About Remote Laboratories Paradigms - Foundations of Ongoing Mutations*, International Journal of Online Engineering, vol. 4, N° 1, pp. 19--25, february 2008.

[7] Gravier, C., Fayolle, J., Noyel, G., Leleve, A., Benmohamed, H., *Distance Learning: Closing the Gap between Remote Labs and Learning Management Systems*, Proceedings of IEEE First International Conference on E-Learning in Industrial Electronics (ICELIE’2006), Hammamet , Tunisia, December 18-20, 2006

[8] Berners-Lee, T., *WWW past & future*, Available at http://www.w3.org/2003/Talks/0922-rsoc-tbl/, 2003, updated in 2007.

[9] Horrocks, I., Parsia, B., Patel-Schneider, P. and Hendler, J., *Semantic Web Architecture: Stack or Two Towers?*, in Francois Fages and Sylvain Soliman, editors, Principles and Practice of Semantic Web Reasoning (PPSWR 2005), number 3703 in Lecture Notes on Computer Science, pp. 37-41. Springer, 2005.

[10] J. Garcia-Zubia, U. Hernández-Jayo, I. Angulo, P. Orduña, *Remote Laboratories based on LXI*, International Journal of Online Engineering, vol. 4, N° 3, pp. 25--27, february 2008.

[11] Gutterman L., *Integrating VISA, IVI and ATEasy to migrate legacy test system*, in Proceedings of AUTOTESTCON’04, 20–23 September, 2004

[12] Landragin J.-P. & Schwartz P., *L’instrumentation PXI*, L’électronique, n°165, January 2006

AUTHORS

C. Gravier, J. Fayolle, J. Lardon, B. Bayard, G. Dusser, R. Vérot are with TELECOM Saint-Etienne, an associate engineering school of the French Institut TELECOM. TELECOM Saint-Etienne is at Université de Saint-Etienne, Université de Lyon, 42000 Saint-Etienne, France (e-mail: {christophe.gravier, jacques.fayolle, bernard.bayard, jeremy.lardon }@telecom-st-etienne.fr).

This work was supported by the General Council of Loire, France.

This article was modified from a presentation at the Workshop on Remote control of Devices (WoRD 2008) in London, United Kingdom, November 13-16, 2008. Manuscript received 30 January 2009. Published as submitted by the authors.