# Designing a Virtual Laboratory for a Relational Database MOOC

Olivier Berger[1], J. Paul Gibson[1,2], Claire Lecocq[1,2] and Christian Bac[1]
[1]*Institut Mines-Télécom, Télécom SudParis, Évry, France*
[2]*UMR CNRS 5157 SAMOVAR, Évry, France*

Keywords: Remote Learning, Virtualization, Open Education Resources, MOOC, Vagrant.

Abstract: Technical advances in machine and system virtualization are creating opportunities for remote learning to provide significantly better support for active education approaches. Students now, in general, have personal computers that are powerful enough to support virtualization of operating systems and networks. As a consequence, it is now possible to provide remote learners with a common, standard, virtual laboratory and learning environment, independent of the different types of physical machines on which they work. This greatly enhances the opportunity for producing re-usable teaching materials that are actually re-used. However, configuring and installing such virtual laboratories is technically challenging for teachers and students. We report on our experience of building a virtual machine (VM) laboratory for a MOOC on relational databases. The architecture of our virtual machine is described in detail, and we evaluate the benefits of using the Vagrant tool for building and delivering the VM.

## 1 INTRODUCTION

Teaching Computer Science requires significant practical experimentation by the students, yet — contrary to other disciplines like physics, electronics or life sciences — specialised hardware devices are not generally required. This is fortunate in the context of distance learning, as lack of accessibility to dedicated hardware is not necessarily a major problem. Even so, remote CS laboratories are non-trivial to develop due to the heterogeneity of the equipment, and the diversity of competences of remote learners,

After teaching an introductory *Relational Databases Course* for more than a decade, we have delivered the first edition of a similar course in the form of a MOOC In addition, a laboratory environment based on VM technology was developed to allow distant learners to practice on concrete Relational Database tools. This paper describes the use of the Vagrant (Hashimoto, 2013) technology for building and delivering such a VM-based laboratory.

### 1.1 A Brief History of Distance Learning

The evolution of distance learning has always been tightly coupled to the evolution of communication technologies. In *Serving the system: A critical history of distance education* (Sumner, 2000), the first three generations are reviewed: (1) correspondence courses, (2) multimedia courses, and (3) computer conferencing. In1994, we see the emergence of the fourth generation of remote learning, when James O'Donnell pioneered the use of the internet for remote teaching (Brown and Duguid, 1996). The last 20 years have seen this 4th generation evolve to address the issues of openness and scalability, starting in 2007 with *Rhizo14*, the first MOOC (Cormier, 2014). We believe that the future of on-line learning is likely to be dominated by MOOLs (Massively Open On-Line Laboratories) (Lowe, 2014), whose success will be wholly dependent on the supporting virtualization technologies.

We are already evolving towards the next generation of remote learning. A key driving force is the growth in VMs of different kinds: from virtual operating systems (Chen and Noble, 2001) to virtual infrastructure, networks and services on the cloud (Armbrust et al., 2010).

### 1.2 Virtualization: The Challenges

There are a number of outstanding remote educational challenges that virtualization will help us to address. There are also a number of new challenges that virtualization will introduce:

**1) The Active-learning Challenge** - In the last


Berger O., Gibson J., Lecocq C. and Bac C..
Designing a Virtual Laboratory for a Relational Database MOOC.
DOI: 10.5220/0005439702600268
In *Proceedings of the 7th International Conference on Computer Supported Education* (CSEDU-2015), pages 260-268
ISBN: 978-989-758-107-6

decade there has been increasing recognition of the importance of active learning (Prince, 2004). Yet, all previous generations of remote learning have suffered from the difficulty of students not being able to work on practical projects in a controlled environment. This has led to the some disciplines, generally with an engineering focus, developing virtual laboratories which can be controlled remotely (Ma and Nickerson, 2006). However, this approach is extremely resource intensive and lacks flexibility.

**2) The Adaptive-learning Challenge** - A second issue that needs to be addressed is that of adaptive learning (Jones and Winne, 2012). For on-line courses with a large number of participants, adaptive learning can be achieved, with a reasonable use of human resources, only if there is automated support. However, when using VMs, the teacher may no longer have complete control over the laboratory and so may not be able to access the required information.

**3) The Technology Challenge** - Students and teachers struggle with the adoption of unfamiliar technology, and this is even more of an issue with on-line courses. VMs (running on the cloud) have been used in specialist computer courses concerned with networks and security for more than a decade (Cooper, 2005); but the problems reported by early adopters of such systems continue to be an issue today (Winckles et al., 2011).

**4) The Re-usability Challenge** - Re-use of educational resources is a significant challenge for laboratory-based courses. MOOLs follow the MOOC philosophy with focus on the scaleability of remote practical work and experimentation (Lowe, 2014), rather than on reusability of material across different courses.

**5) The Integration/standardisation Challenge** - The integration of the VM (or laboratory) with the learning management and content management systems is critical (Gomes and Bogosyan, 2009) . Further, there is a need for standardisation of the system interfaces in order for this integration to be done in a rigorous manner.

**6) The Internet Challenge** - Based on the principle of equality, the teacher makes the assumption that the students are all working on the same VM/system. These systems will depend more-or-less on the internet for downloading support material. This raises issues with respect to security, anonymity, cost of access, dependability, openness and intellectual property (Pearson, 2013)

**7) The Deployment Challenge** - Once the teacher has a laboratory session designed to work with a particular VM then there is still the major problem of deploying the specific machine on each student's personal computer. In general, this can be broken down into three steps: ensuring that the student's computer supports the virtualization, configuring the specific VM for the particular session, and supporting the student during the laboratory session. The third deployment stage is, perhaps, the most challenging. Providing such support has proven to be fundamental for the success of MOOCs (Alario-Hoyos et al., 2014) and successful deployment of a virtual laboratory will most certainly depend on the support mechanisms.

**8) The Scalability Challenge** - A main goal of online open education is to reach the masses. A typical laboratory session is limited to a relatively small number of students per class. With a virtual laboratory, one would hope that no such constraints exist. In theory, there would be no limit to the number of students that could download the VM and follow the laboratory session. However, there are certain key resources that are not limitless: the teacher(s) and technicians are limited by the amount of time and equipment that they can spend on providing the necessary support for the VMs (and their deployment).

### 1.3 The Design Problem

Each of the previously identified challenges is worthy of consideration in its own right. However, developing a system that meets all of the challenges requires research into the different possible designs that current technology would support and the associated compromises that would be required. Our initial approach to this problem is quite pragmatic and consists of trying to verify whether Vagrant (Hashimoto, 2013), a modern virtualization framework tool popular in the development communities, helps address these challenges.

## 2 THE VIRTUALIZATION REQUIREMENTS

In current laboratory education, there are three main actors in the system: educators, laboratory technicians, and learners. When replacing a concrete laboratory with a virtual laboratory, our goal is to improve the system for each of these different types of actor. In the following section, we outline a scenario-driven approach to identifying the main requirements.

### 2.1 Scenario-based Requirements

Initially, we consider the typical pre-virtualization scenario, as a sequence of 4 stages —

**1) Educator-Driven Experimentation:** First, a teacher will have identified a number of learning objectives that the session must fulfil. Then, the teacher will outline the session, and experiment with the tools that the students will be expected to use. After many iterations, the teacher will be satisfied that the content is appropriate for the learning objectives and that the laboratory set-up is satisfactory.

**2) Technician-Driven Deployment:** Before each laboratory class begins, a technician will be required to set up the laboratory tools as specified by the teacher. The main objective is to ensure that each student will use a laboratory environment equivalent to that which the teacher finalised in the previous step.

**3) Student-Driven Learning:** The students are required to be physically present when the laboratory session is being run. They can expect the equipment and tools to work as required, and any problems can be resolved easily during the session. They can focus on the course content and meeting the learning objectives. During the class, students can get help from the teacher; and this can often take the form of the teacher demonstrating the use of the equipment, or even helping the student to accomplish certain tasks. During the class, the teacher has the opportunity to interact and observe the behaviour of all participants.

**4) Feedback:** After class, the teacher may also get feedback from students and technicians regarding the set up of the laboratory tools. Using this feedback, the teacher can improve the session for the next class (or year). The session can also be published for reuse by other colleagues and institutions.

A virtual laboratory must support the same type of scenario without adding extra burden on the three main actors. Teachers must be able to easily experiment with the laboratory set-up, and expect any set-up to be reproducible by the technician. The technician must be able to understand the teacher's requirements and be able to easily reproduce a set-up. Students must have easy access to the laboratory. There must be a range of feedback mechanisms in place. Further, students need to be supported during their laboratory work. This support needs to be interactive and context sensitive. Finally, none of the three actors in the system should be expected to undertake any additional risk through participating in this process.

Within the specific context of VMs for learning, we transform these informal scenario-based requirements into a set of criteria against which any VM laboratory can be judged: **1) Ease of configuration** and experimentation (for the educator), **2) Ease of specification** and validation (for the educator and the technician), **3) Reliability** (for all 3 actors), **4) Safety** (for all 3 actors), **5) Cost** (for all 3 actors), **6) Real-time support** and monitoring (for learners) and **7) Feedback** from data gathering (for all 3 actors)

Table 1 – Comparing Scalability of Teaching Laboratories.

| Lab. Type | 1 Ease of configuration | 2 Ease of specification | 3 Reliability | 4 Safety | 5 Cost | 6 Real-time support | 7 Feedback |
|---|---|---|---|---|---|---|---|
| **Traditional** | - | - | +/- | +/- | - | + | +/- |
| **Simulated** | - | +/- | + | + | - | +/- | + |
| **Cloud-based VM** | + | + | - | + | - | +/- | +/- |
| **Local VM** | + | + | + | + | + | - | +/- |

## 2.2 Related Work on Requirements

Previous research has also identified these criteria as being major issues. For example, the importance of feedback helps the teacher to verify that students are using tools in a rigorous manner (Billingsley and Steel, 2014). In a remote networks course (Bullers et al., 2006), we see the benefits and challenges in course and lab configuration, security, and administration when using VMs. The challenge of real-time monitoring of students is shown to be strongly related to the problem of integrating the different tools (Romero-Zaldivar et al., 2012). The interoperability issue is also critical for standardisation (Yeung et al., 2010).

## 2.3 Scalability of Existing Approaches

In the following, we compare the different laboratory environments with respect to the criteria identified, above, when deployed for a typical computing laboratory session, such as for our relational databases course. In table 1, we annotate positive aspects as '+', and negative aspects as '-'. Where there is no clear positive or negative tendency we note this as '+-'.

It should be noted that our evaluation is based on the assumption that our primary goal is to achieve a laboratory environment that is scalable. So, for example, a traditional laboratory may be easy to configure for a small number of students but configuration will usually pose significant problems for a large number of students.

# 3 THE MOOC LABORATORY

The first edition of our MOOC on Relational Databases was hosted by Telecom SudParis (TSP) from September to November 2014[1]. This course, taught in French, reused existing content and tools of a course that had already been taught for a decade to 2nd year students in the engineering curriculum The course has gathered a peak of 1950 registered participants (including 180 students from our own institute), from around 20 countries (mostly Francophone) with 75% from France and around 15% in Africa. The MOOC was delivered on our own local installation of Moodle (Dougiamas and Taylor, 2003), mixing the typical resources: videos, PDF files, quizzes and links to download supporting tools for exercises and labs.

## 3.1 Exercises and Lab Tools

For the needs of the course, our intention was to let participants experiment with the following tools:
**1) A standalone SQL queries tutor environment** available for the Safari or Chrome Web browser. This tutor environment helps students compose queries in the SQL language in a Web form. Helper widgets offer an easy access to keywords of the language or names of database artefacts extracted from the schema of the example databases (names of tables or columns). This Javascript application[2] relies on the embedded SQLite database manager, but which is only available in the WebKit engine integrated in the aforementioned Web browsers. Thus, even if it doesn't require us to install a real RDBMS to operate, it may impose an inconvenience on the user by requiring the use of a particular Web browser.
**2) An alternative version of the SQL queries tutor application**, with a similar interface to the standalone variant mentioned above, but implemented in the form of an HTML + Javascript and PHP application (plugged on a RDBMS). This tool has been developed over a number of years at our institute, and used to be operated on a server hosted by the IT services. Contrary to the previous approach, the main advantage of this variant is the compatibility with any Web browser, but in turn it requires a working Web server and RDBMS, which may be hard to install locally.
**3) A professional grade Relational Database Management System (RDMBS)**, namely PostgreSQL[3], already populated with example databases, on to which queries will be performed.
**4) The phpPgAdmin application**[4], a database query and management tool adapted to Postgresql, requiring a local Web server with PHP.
**5) Other laboratory material**, in the form of some Database and Web programming PHP program templates, serving in the introduction to Web forms and database query programming in PHP. The accompanying PHP engine (also necessary for phpPgAdmin or the tutor application) would typically be installed over the Apache Web server.

## 3.2 From Requirements to Design

In the context of the MOOC, the availability of an integrated laboratory as a VM (VM) would directly address the following criteria from our requirements:

1. **Ease of configuration -** participants should focus on the use of the applications made available to them in the VM, without the need to learn how to install or configure them. No *sysadmin* competence should be a prerequisite of the MOOC.
2. **Ease of specification -** we automate the production of the VM, through the use of building scripts. This ensures the maintainability of the resulting VM (updates, reproducibility, auto-documented, etc.).
3. **Reliability -** the labs can be performed off-line: there is no need to access particular (Cloud) servers. Further, there is no need for high bandwidth internet connection during the laboratory sessions. Of course this goal is mitigated by the need to perform an initial download of a VM image (although we'd try to make it as minimal as possible). Reliability is also improved by a more direct delivery cycle: the most recent VM tested during the teacher's experimentation, and specified for deployment, should be exactly the same as the one that will be made available to the MOOC participants.
4. **Safety -** with our VM approach, the educator is protected against the possibility of installing dangerous software on his own computer. The students are also similarly protected. Secondly, the technician does not have to worry about the possibility of students breaking the shared computing resources whilst following a lab session. Finally, the students are protected from having to trust a third party (other than their own academic institute) with respect to confidentiality.

[1] See http://mooc.telecom-sudparis.eu/mod/book/view.php?id=431
[2] See https://fusionforge.int-evry.fr/projects/envmoocbd
[3] See http://www.postgresql.org/
[4] See http://phppgadmin.sourceforge.net/

5. **Cost -** with the use of free software, no extra fees should be required to participate in the MOOC. Also, given that the MOOC resources will be released under a Creative Commons open license, the accompanying tools should be available, to the maximum extent possible for every potential need of reuse or adaptation of the MOOC;
6. **Real-time support -** Our design recommends real-time support, but we have yet to incorporate this functionality in our system prototype, which isn't obvious given that the VMs may be used off-line. This requires further research into the use of intelligent tutoring and help systems.
7. **Feedback -** Our design recommends monitoring and gathering of data (on-line and off-line), but we have yet to incorporate this functionality in our system prototype. Using a cloud-based VM is very likely to impact on the type of feedback that could be generated (and the confidentiality of the data); however, using a local VM offers many opportunities for automated gathering of *big data* to help in profiling of individual students, as well as the whole class.

# 4 MAKING THE VM AS A VAGRANT BOX

We have chosen to experiment with Vagrant(Hashimoto, 2013) and VirtualBox(Watson, 2008) to build, distribute and finally run the Vitual Machine.

Vagrant mainly provides scriptability for the generation of the box and VirtualBox ensures it is portable on most operating systems. In the following, we will refer to the VM, using the Vagrant jargon which refers to a VM image as a *box*.

Figure 1 which illustrates the process of building a box with Vagrant. The following sequence is followed during the build:

1. When the `vagrant up` command is executed, Vagrant climbs the directory tree looking for the first `VagrantFile` it can find. It loads this file into a merged sequence of Vagrant files (found in various locations, allowing for different priorities and overriding).
2. The final merged `Vagrantfile` contains a number of `Vagrant.configure` blocks that are used to configure the scripts and resources to be used in the build.
3. In our configuration, these resources are links to the Debian packages archive and the Git repository for the laboratory applications.
4. The output of the configuration process is the Vagrant box that can be re-used by any Vagrant environment.

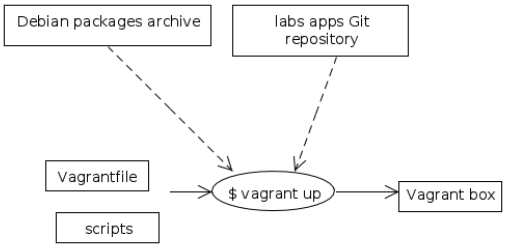


Figure 1 – Initial provisioning to build the VM/box.

The OS we have selected for running inside the VM is the Debian (8.0 aka *jessie*) Linux distribution, known for its stability and packaging system. The box contains a minimal Debian system which executes (in the background) PostgreSQL, Apache + `mod_php`, phpPgAdmin, and a few in-house applications of our own. Example databases are already populated in the PostgreSQL RDBMS.

The VMs are intended to be run *headless* for the moment, keeping their size to the minimum, even though we also provide a script to install and configure a desktop environment based on XFCE4.

The exercises of the MOOC section on PHP + SQL programming can be performed using a subfolder of the shared `/vagrant/` folder (a standard feature offered by Vagrant). This shared folder allows editing of PHP or HTML source files on the host system, with the favourite native editor/IDE, while running in place the PHP scripts inside the VM?s Apache + `mod_php`.

## 4.1 Portability Issues

The majority of the personal computers used nowadays are running with either 32 bit or 64 bit CPUs, which may correspond to a dedicated version of the Operating System. It is worth noting that, in general, 64 bit CPUs can run 32 bit applications, while the reverse isn't possible.

To adjust to this limitation of the virtualization technique, we generated 2 variants of the box, one for a 32 bits kernel (`i686`) which can be installed everywhere, and one containing a 64 bits kernel (`amd64`) which must be installed only, and preferably, on 64 bits operating systems.

## 4.2 Delivery Through Internet

The resulting boxes, once automatically compressed by Vagrant, are between 300 and 350 Mb in size.

They are uploaded to a self-hosting site at our local institution, and distributed through the catalogue at `vagrantcloud.com`. Once the VM are created in VirtualBox, the typical VMDK drive images file is around 800 Mb and typically grows up to 1.3Gb.

Figure 2 gives a simplified overview of the downloading of the box and final provisioning of the VM in VirtualBox, that is performed on the participant's computer. The sequence is made up of 3 main steps:

1. When the `vagrant init` command is executed, Vagrant initialises the current directory to be a Vagrant environment, preconfigured to an already existing box (if specified)
2. When the `vagrant up` command is executed, Vagrant links to the specified resources at the cloud repository and the virtual box on the web server.
3. The final output is a virtual disk image that can be use in VirtualBox.

Note that we also included a mechanism (not shown in the diagram) allowing for last minute or on-demand updates of the laboratory materials. This works by updating from a published Git repository, which allows us to distribute corrections to the lab materials without having to regenerate a full image of the VM.

### 4.3 Security

We use our own Debian base boxes containing a minimal Debian jessie/testing, instead of relying on a secondary source. This ensures we can put more trust in the content as it is a native Debian package installation without any man-in-the-middle intervention, and is deemed best for our MOOC participants.

### 4.4 Availability of the Box Sources

The sources of the whole project (scripts, Vagrantfiles, docs, etc.) are available as free software[5].

for anyone interested in replicating a similar environment for another project.

## 5 VALIDATION

Given the moderate effort involved in setting up the MOOC lab, we're very much satisfied by the initial results, both from the point of view of the professors and engineers who made the VM available to the MOOC participants, or judging by the feedback received from the majority of MOOC participants that used the VM for the labs.

Clearly, the main benefit is on the ability to assemble and test a coherent environment comprised of multiple components, in a reproducible and documented manner,. This environment is customized in a predefined manner, while using "real" professional tools, which allow controlled and realistic experiments, without the need to develop complex simulation tools. We benefit from the stability and maturity of standard tools like PostgreSQL, without putting the additional burden of learning how to configure it on the students.

### 5.1 Reliability Issues with VirtualBox

The VirtualBox virtualization engine works quite well in the majority of the cases, but sometimes the VMs won't start (or start won't be detected correctly), and then, there's very few diagnostic elements to be able to help participants: remote debugging through MOOC forums isn't easy.

Most of the issues were reported by Windows users (and seemed related to network or firewall issues). Note that most of the situations reported were also already frequently reported in public forums for other cases of VM use, and thus are likely bugs or misconfigurations in the VirtualBox technology and not related to our specific use of Vagrant and VirtualBox.

In the face of such difficulties, some participants have preferred to install PostgreSQL distributions on their systems. They then had to create and populate the databases by themselves, which wasn't so much a problem for most cases, given we provided them with links to our scripts available for download.

### 5.2 Student Feedback and Evaluation

Although this paper is focused on our VM architecture as a proof of concept, we have collected some relevant data with regards to our initial experimentation. It should be noted that ongoing and future use of the VM architecture will focus more on the non-functional requirements, and the means by which we can improve the architecture in order to facilitate the collection and analysis of usage data. It is too early in our research to fully validate all of our requirements, but our initial evaluation demonstrates that the system functions as required in the majority of cases.

The total number of learners involved[6] in the

[5] Available at https://fusionforge.int-evry.fr/projects/moocbdvm/

[6] These were learners who had a significant participation but did not necessarily cover all of the material.

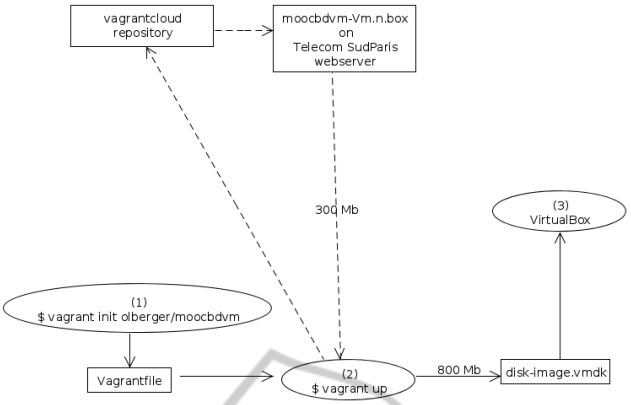


Figure 2 – Simplified flow of the final provisioning to launch a new VM instance in VirtualBox.

MOOC was more than 1900. We did not have a means to count the number of VM downloads, but estimate — based on informal student feedback and analysis of the forum discussions — that between 5% and 10% of the learners installed and used the VM laboratory.

From these downloads we estimate that 30% were Windows users. From an estimated total of 40, around half of them reported issues. Further analysis showed that these issues were related to 4 distinct problems. Approximately 75% of the learners who reported problems required assistance in resolving them. We have no data concerning learners who had problems but did not report them.

With regards to the option of installing PostgreSQL, approximately 10% of the learners made this choice. We are investigating the reason behind this decision, as it may signify that they had experienced (as yet unreported) problems with the other options.

Overall, our initial validation of the system has demonstrated the feasibility of the approach, the correct functioning of the system for a significant majority of users, and the need for more rich data collection mechanisms. In the next version and deployment of the system, we plan to gather the usage data in an automated fashion, with focus on problem identification and resolution techniques.

# 6 FUTURE WORK

1noindent Rather than reacting to advances in virtualization technology, educators should — as one of the most important communities of potential users — be involved in directing the technology through research into how best it can be used. We propose 2 key areas for such future research, which will also address the limitations we observed in our prototype, with respect to requirements initially set.

## 6.1 Laboratory Monitoring

One of the drawbacks of the use of so called Vagrant *boxes* and more generally VMs, is that they tend to operate like *black boxes*, from the pedagogical point of view. This is clearly suboptimal with respect to the "real-time support" (6) or "feedback" (7) requirements we have defined (see 3.2).

Isolation of operation is a necessary condition for a VM system: applications running inside don't interfere with the physical machine over which they are run. Yet, such isolation is not desirable if we wish to fully monitor the learner's actions during a lab. Furthermore, it is then very difficult to provide feedback to guide learners as they are learning We envision a way to solve this "tunnel effect", and avoid the laboratory activities occurring below the radar of the LMS, while keeping all the benefits of the virtualization technique. This could take the form of a middleware component that would be embedded inside the VM, and would facilitate monitoring certain actions of the participant, by collecting selected traces, and pushing them to the LMS once the machine is connected back to the Internet.

## 6.2 More Modular VMs

We envision other ways to improve the VMs, which could help to address the needs of several courses or modular courses that would make use of a single VM environment. The system inside the VM would contain several bricks corresponding to different sections of a course, or different courses. Instead of a quite monolithic VM/box, the participants could then download variable sets of lab components that would integrate on a common base.

Such an architecture could be beneficial in the context of alternative deployments on Cloud plat-

forms. The same building blocks of individual labs could be assembled to deliver both VMs / boxes, or Cloud instances. The former would suit best the MOOC distant/disconnected learners, and the latter, some on-site labs and classes, or cases when a learner doesn't own a powerful enough device to run virtualized systems.

## 7 CONCLUSIONS

The next generation of remote learning technologies must address the issue of providing open, scaleable and sustainable educational resources that support these types of learning. The current generation of remote teaching can be adapted to these types of learning in cases where the students are not required to work with a specific concrete machine or within a specific environment. However, in many cases — particularly in the science and engineering disciplines — this is not possible. virtualization holds the key to solving this problem.

The Vagrant prototype architecture that we report on in this paper has demonstrated the feasibility of VM-based MOOLs (massive open online laboratories). Even if we achieved a good conformance to the requirements for such a laboratory, more research needs to be done, particularly on the real-time feedback provided to students, who may be working offline; and on the collection and exploitation of the huge amounts of data that could be collected on the students' machines.

## ACKNOWLEDGEMENTS

A big thanks go to our intern Stéphane Germain, who joined us during the summer 2014 to work on this virtualized environment. This experiment was conducted with financial support from Idex Paris-Saclay (as part of the « Former avec le numérique dans l'Université Paris-Saclay 2014 » programme).

## REFERENCES

Alario-Hoyos, C., Pérez-Sanagustín, M., Kloos, C. D., and Muñoz Merino, P. J. (2014). Recommendations for the design and deployment of MOOCs: Insights about the MOOC digital education of the future deployed in MiríadaX. In *Proceedings of the Second International Conference on Technological Ecosystems for Enhancing Multiculturality*, TEEM '14, pages 403–408, New York, NY, USA. ACM.

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., and Zaharia, M. (2010). A view of cloud computing. *Commun. ACM*, 53:50–58.

Billingsley, W. and Steel, J. R. (2014). Towards a supercollaborative software engineering MOOC. In *Companion Proceedings of the 36th International Conference on Software Engineering*, pages 283–286. ACM.

Brown, J. S. and Duguid, P. (1996). Universities in the digital age. *Change: The Magazine of Higher Learning*, 28(4):11–19.

Bullers, Jr., W. I., Burd, S., and Seazzu, A. F. (2006). Virtual machines - an idea whose time has returned: Application to network, security, and database courses. *SIGCSE Bull.*, 38(1):102–106.

Chen, P. M. and Noble, B. D. (2001). When virtual is better than real [operating system relocation to virtual machines]. In *Hot Topics in Operating Systems, 2001. Proceedings of the Eighth Workshop on*, pages 133–138. IEEE.

Cooper, M. (2005). Remote laboratories in teaching and learning–issues impinging on widespread adoption in science and engineering education. *International Journal of Online Engineering (iJOE)*, 1(1).

Cormier, D. (2014). Rhizo14–the MOOC that community built. *INNOQUAL-International Journal for Innovation and Quality in Learning*, 2(3).

Dougiamas, M. and Taylor, P. (2003). Moodle: Using learning communities to create an open source course management system. In *World conference on educational multimedia, hypermedia and telecommunications*, pages 171–178.

Gomes, L. and Bogosyan, S. (2009). Current trends in remote laboratories. *Industrial Electronics, IEEE Transactions on*, 56(12):4744–4756.

Gu, Q. and Sumner, T. (2006). Support personalization in distributed e-learning systems through learner modeling. In *Information and Communication Technologies, 2006. ICTTA'06. 2nd*, volume 1, pages 610–615. IEEE.

Hashimoto, M. (2013). *Vagrant: Up and Running*. O'Reilly Media, Inc.

Jones, M. and Winne, P. H. (2012). *Adaptive Learning Environments: Foundations and Frontiers*. Springer Publishing Company, Incorporated, 1st edition.

Lowe, D. (2014). MOOLs: Massive open online laboratories: An analysis of scale and feasibility. In *Remote Engineering and Virtual Instrumentation (REV), 2014 11th International Conference on*, pages 1–6. IEEE.

Ma, J. and Nickerson, J. V. (2006). Hands-on, simulated, and remote laboratories: A comparative literature review. *ACM Computing Surveys (CSUR)*, 38(3):7.

Pearson, S. (2013). Privacy, security and trust in cloud computing. In *Privacy and Security for Cloud Computing*, pages 3–42. Springer.

Prince, M. (2004). Does active learning work? A review of the research. *Journal of engineering education*, 93(3):223–231.

Romero-Zaldivar, V.-A., Pardo, A., Burgos, D., and Delgado Kloos, C. (2012). Monitoring student progress

using virtual appliances: A case study. *Computers & Education*, 58(4):1058–1067.

Smith, J. E. and Nair, R. (2005). The architecture of virtual machines. *Computer*, 38(5):32–38.

Sumner, J. (2000). Serving the system: A critical history of distance education. *Open learning*, 15(3):267–285.

Watson, J. (2008). Virtualbox: Bits and bytes masquerading as machines. *Linux J.*, 2008(166).

Winckles, A., Spasova, K., and Rowsell, T. (2011). Remote laboratories and reusable learning objects in a distance learning context. *Networks*, 14:43–55.

Yeung, H., Lowe, D. B., and Murray, S. (2010). Interoperability of remote laboratories systems. *iJOE*, 6(S1):71–80.