A review of Virtual Labs virtualization solutions for MOOCs

From Virtual Machines running locally or on IaaS, to containers on a PaaS, up to hypothetical ports of tools to WebAssembly for serverless execution in the Web browser

Olivier Berger, Télécom SudParis <olivier.berger@telecom-sudparis.eu>

February 9, 2018

Contents

1 About this document

A copy of this memo was published simultaneously at the author's website, *where the reader may find the most up-to-date version*

2 Introduction

This is a memo that tries to capture some of the experience gained in the FLIRT project on the topic of Virtual Labs for MOOCs (*Massive Open Online Courses*).

In this memo, we try to draw an overview of some benefits and concerns with existing approaches at using virtualization techniques for running Virtual Labs, as distributions of tools made available for distant learners.

We describe 3 main technical architectures: (1) running Virtual Machine images locally on a virtual machine manager, or (2) displaying the remote execution of similar virtual machines on a IaaS cloud, and (3) the potential of connecting to the remote execution of minimized containers on a remote PaaS cloud.

We then elaborate on some perspectives for locally running ports of applications to the WebAssembly virtual machine of the modern Web browsers.

Disclaimer: *This memo doesn't intend to point to extensive literature on the subject, so part of our analysis may be biased by our particular context.*

3 Context : MOOCs

Many MOOCs (*Massive Open Online Courses*) include a kind of "virtual laboratory" for learners to experiment with tools, as a way to apply the knowledge, practice, and be more active in the learning process. In quite a few (technical) disciplines, this can consist in using a set of standard applications in a professional domain, which represent typical tools that would be used in real life scenarii.

Our main perspective will be that of a MOOC editor and of MOOC production teams which want to make "virtual labs" available for MOOC participants.

Such a "virtual lab" would typically contain installations of existing applications, pre-installed and configured, and loaded with scenario data in order to perform a lab.

The main constraint here is that such labs would typically be fabricated with limited software development expertise and funds[1]. Thus we consider here only the assembly of existing "normal" applications and discard the option of developping novel "serious games" and simulator applications for such MOOCs.

3.1 The FLIRT project

The FLIRT project groups a consortium of 19 partners in Industry, SMEs and Academia to work on a collection of MOOCs and SPOCs for professional development in Networks and Telecommunications. Lead by Institut Mines Telecom, it benefits from the funding support of the French "*Investissements d'avenir*" programme.

As part of the FLIRT roadmap, we're leading an "innovation task" focused on Virtual Labs in the context of the Cloud. This memo was produced as part of this task.

3.2 Some challenges in virtual labs design for distant learning

Virtual Labs used in distance learning contexts require the use of software applications **in autonomy**, either running on a personal, or professional computer. In general, the technical skills of participants may be diverse. So much for the quality (bandwith, QoS, filtering, limitations: firewaling) of the hardware and networks they use at home or at work. It's thus very optimistic to seek for *one solution fits all* strategy.

Most of the time there's a learning curve on getting familiar with the tools which students will have to use, which constitutes as many challenges to overcome for beginners. These tools may not be suited for beginners, but they will still be selected by the trainers as they're representative of the professional context being taught.

In theory, this usability challenge should be addressed by devising an adapted pedagogical approach, especially in a context of distance learning, so that learners can practice the labs on their own, without the presence of a tutor or professor. Or some particular prerequisite skills could be required ("please follow *System Administration 101* before applying to this course").

Unfortunately there are many cases where instructors basically just translate to a distant learning scenario, previous lab resources that had previously been devised for in presence learning. This lets learner faced with many challenges to overcome. The only support resource is often a regular forum on the MOOC's LMS (*Learning Management System*).

My intuition[2] is that developing ad-hoc *simulators* for distant education would probably be more efficient and easy to use for learners. But that would require a too high investment for the designers of the courses.

In the context of MOOCs which are mainly free to participate to, not much investment is possible in devising ad-hoc lab applications, and instructors have to rely on existing applications, tools and scenarii to deliver a cheap enough environment. Furthermore, technical or licensing constraints[3] may lead to selecting lab tools which may not be easy to learn, but have the great advantage or being freely redistributable[4].

4 Virtual Machines for Virtual Labs

The learners who will try unattended learning in such typical virtual labs will face difficulties in making specialized applications run. They must overcome the technical details of downloading, installing and

[1] The FLIRT project also works on business model aspects of MOOC or SPOC production in the context of professional development, but the present memo starts from a minimalitic hypothesis where funding for course production is quite limited.

[2] research-based evidence needed

[3] In typical MOOCs which are free to participate, the VM should include only gratis tools, which typically means a GNU/Linux distribution loaded with applications available under free and open source licenses.

[4] Typically, Free and Open Source software, aka Libre Software

configuring programs, before even trying to perform a particular pedagogical scenario linked to the matter studied.

To diminish these difficulties, one traditional approach for implementing labs in MOOCs has been to assemble in advance a **Virtual Machine image**. This already made image can then be downloaded and run with a virtual machine simulator (like VirtualBox[5]).

The pre-loaded VM will already have everything ready for use, so that the learners don't have to install anything on their machines.

An alternative is to let learners download and install the needed software tools themselves, but this leads to so many **compatibility issues** or technical skill prerequisites, that this is often not advised, and mentioned only as a fallback option.

4.1 Downloading and installation issues

Experience shows[2] that such virtual machines also bring some issues. Even if installation of every piece of software is no longer required, learners still need to be able to run the VM simulator on a wide range of diverse hardware, OSes and configurations. Even managing to download the VMs, still causes many issues (lack admin privileges, weight vs download speed, memory or CPU load, disk space, screen configurations, firewall filtering, keayboard layout, etc.).

These problems aren't generally faced by the majority of learners, but the impacted minority is not marginal either, and they generally will produce a lot of **support requests** for the MOOC team (usually in the forums), which needs to be anticipated by the community managers.

The use of VMs is no show stopper for most, but can be a serious problem for a minority of learners, and is then no silver bullet.

Some general **usability** issues may also emerge if users aren't used to the look and feel of the enclosed desktop. For instance, the VM may consist of a GNU/Linux desktop, whereas users would use a Windows or Mac OS system.

4.2 Fabrication issues for the VM images

On the MOOC team's side, the fabrication of a lightweight, fast, tested, license-free and easy to use VM image isn't necessarily easy.

Software configurations tend to rot as time passes, and maintenance may not be easy when the later MOOC editions evolutions lead to the need to maintain the virtual lab scenarii years later.

Ideally, this would require adopting an "industrial" process in building (and testing) the lab VMs, but this requires quite an expertise (system administration, packaging, etc.) that may or not have been anticipated at the time of building the MOOC (unlike video editing competence, for instance).

Our experiment with the Vagrant technology [0] and Debian packaging was interesting in this respect, as it allowed us to use a well managed "script" to precisely **control the build** of a minimal VM image.

5 Virtual Labs as a Service

To overcome the difficulties in downloading and running Virtual Machines on one's local computer, we have started exploring the possibility to run these applications in a kind of Software as a Service (SaaS) context, "on the cloud".

But not all applications typically used in MOOC labs are already available for remote execution on the cloud (unless the course deals precisely with managing email in GMail).

We have then studied the option to use such an approach not for a single application, but for a whole **virtual "desktop"** which would be available **on the cloud**.

5.1 IaaS deployments

A way to achieve this goal is to deploy Virtual Machine images quite similar to the ones described above, on the cloud, in an *Infrastructure as a Service* (IaaS) context[6], to offer access to remote desktops for every learners.

There are different technical options to achieve this goal, but a simplified description of the architecture can be seen as just running Virtual Machines on a single IaaS platform instead of on each learner's computer. Access to the desktop and application interfaces is made possible with the use of Web pages

[5] VirtualBox is portable on many operating systems, making it a very popular solution for such a need

[6] the IaaS platform could typically be an open cloud for MOOCs or a private cloud for SPOCs (for closer monitoring of student activity or security control reasons).

(or other dedicated lightweight clients) which will display a "full screen" display of the remote desktop running for the user on the cloud VM. Under the hood, the remote display of a Linux desktop session is made with technologies like VNC and RDP connecting to a Guacamole server on the remote VM.

In the context of the FLIRT project, we have made early experiments with such an architecture. We used the CloVER solution by our partner ProCAN which provides a virtual desktops broker between OpenEdX and an OpenStack IaaS public platform.

The expected benefit is that users don't have to install anything locally, as the only tool needed locally is a **Web browser** (displaying a full-screen HTML5 canvas displaying the remote desktop run by the Guacamole server running on the cloud VM.

But there are still some issues with such an approach. First, the cost of operating such an infrastructure : Virtual Machines need to be hosted on a IaaS platform, and that cost of operation isn't null[7] for the MOOC editor, compared to the cost of VirtualBox and a VM running on the learner's side (basically zero for the MOOC editor).

Another issue, which could be more problematic lies in the need for a **reliable connection** to the Internet during the whole sequences of lab execution by the learners[8]. Even if Guacamole is quite efficient at compressing rendering traffic, some basic **connectivity** is needed during the whole Lab work sessions, preventing some mobile uses for instance.

One other potential annoyance is the potential delays for making a VM available to a learner (provisioning a VM), when huge VMs images need to be copied inside the IaaS platform when a learner connects to the Virtual Lab activity for the first time (several minutes delays). This may be worse if the VM image is **too big** (hence the need for optimization of the content[9]).

However, the fact that all VMs are running on a platform under the control of the MOOC editor allows new kind of features for the MOOC. For instance, learners can submit results of their labs directly to the LMS without the need to upload or copy-paste results manually. This can help **monitor progress** or perform evaluation or grading.

The fact that their VMs run on the same platform also allows new kinds of **pedagogical scenarii**, as VMs of multiple learners can be interconnected, allowing cooperative activities between learners. The VM images may then need to be instrumented and deployed in particular configurations, which may require the use of a dedicated broker like CloVER to manage such scenarii.

For the records, we have yet to perform a rigorous benchmarking of such a solution in order to evaluate its benefits, or constraints given particular contexts. In FLIRT, our main focus will be in the context of SPOCs for professional training (a bit different a context than public MOOCs).

Still this approach doesn't solve the VMs fabrication issues for the MOOC staff. Installing software inside a VM, be it local inside a VirtualBox simulator of over the cloud through a remote desktop display, makes not much difference. This relies mainly on manual operations and may not be well managed in terms of quality of the process (reproducibility, optimization).

5.2 PaaS deployments using containers

Some key issues in the IaaS context described above, are the cost of operation of running full VMs, and long provisioning delays.

We're experimenting with new options to address these issues, through the use of Linux containers running on a PaaS (*Platform as a Service*) platform, instead of full-fleshed Virtual Machines[10].

The main difference, with containers instead of Virtual Machines, lies in the **reduced size** of images, and much **lower CPU load** requirements, as the container remove the need for one layer of virtualization. Also, the deduplication techniques at the heart of some virtual file-systems used by container platforms lead to really **fast provisioning**, avoiding the need to wait for the labs to start.

The traditional making of VMs, done by installing packages and taking a snapshot, was affordable for the regular teacher, but involved manual operations. In this respect, one other major benefit of containers

[7] Depending of the expected use of the lab by learners, this cost may vary a lot. The size and configuration required for the included software may have an impact (hence the need to minimize the footprint of the VM images). With diminishing costs in general this may not be a show stopper. Refer to marketing figures of commercial IaaS offerings for accurate figures. Attention to additional licensing costs if the OS of the VM isn't free software, or if other licenses must be provided for every learners.

[8] The needs for always-on connectivity may not be a problem for professional development SPOCs where learners connect from enterprise networks for instance. It may be detrimental when MOOCs are very popular in southern countries where high bandwidth is both unreliable and expensive.

[9] In this respect, providing a full Linux desktop inside the VM doesn't necessarily make sense. Instead, running applications full-screen may be better, avoiding installation of whole desktop environments like Gnome or XFCE... but which has usability consequences. Careful tuning and testing is needed in any case.

[10] The availability of container based architectures is quite popular in the industry, but has not yet been deployed to a large scale in the context of large public MOOC hosting platforms, to our knowledge, at the time of writing. There are interesting technical challenges which the FLIRT project tries to tackle together with its partner ProCAN.

is the potential for **better industrialization** of the virtual lab fabrication, as they are generally not assembled manually. Instead, one uses a "scripting" approach in describing which applications and their dependencies need to be put inside a container image. But this requires new competence from the Lab creators, like learning the Docker technology (and the OpenShift PaaS, for instance), which may be quite specialized. Whereas Docker containers tend to become quite popular in Software Development faculty (through the "devops" hype), they may be a bit new to other field instructors.

The learning curve to mastering the automation of the whole container-based labs installation needs to be evaluated. There's a trade-off to consider in adopting technology like Vagrant or Docker: acquiring container/PaaS expertise vs quality of industrialization and optimization. The production of a MOOC should then require careful planning if one has to hire or contract with a PaaS expert for setting up the Virtual Labs.

We may also expect interesting **pedagogical benefits**. As containers are lightweight, and platforms allow to "easily" deploy multiple interlinked containers (over dedicated virtual networks), this enables the setup of more realistic scenarii, where each learner may be provided with multiple "nodes" over **virtual networks** (all running their individual containers). This would be particularly interesting for Computer Networks or Security teaching for instance, where each learner may have access both to client and server nodes, to study client-server protocols, for instance. This is particularly interesting for us in the context of our FLIRT project, where we produce a collection of Computer Networks courses.

Still, this mode of operation relies on a **good connectivity** of the learners to the Cloud. In contexts of distance learning in poorly connected contexts, the PaaS architecture doesn't solve that particular issue compared to the previous IaaS architecture.

6 Future server-less Virtual Labs with WebAssembly

As we have seen, the IaaS or PaaS based Virtual Labs running on the Cloud offer alternatives to installing local virtual machines on the learner's computer. But they both require to be **connected** for the whole duration of the Lab, as the applications would be executed on the remote servers, on the Cloud (either inside VMs or containers).

We have been thinking of another alternative which could allow the deployment of some Virtual Labs on the local computers of the learners without the hassles of downloading and installing a Virtual Machine manager and VM image. We envision the possibility to use the infrastructure provided by modern Web browsers to allow running the lab's applications.

At the time of writing, this architecture is still highly experimental. The main idea is to rebuild the applications needed for the Lab so that they can be run in the "generic" virtual machine present in the modern browsers, the WebAssembly and Javascript execution engine.

WebAssembly is a modern language which seeks for maximum portability, and as its name hints, is a kind of assembly language for the Web platform. What is of interest for us is that WebAssembly is portable on most modern Web browsers, making it a very interesting platform for **portability**.

Emerging toolchains allow recompiling applications written in languages like C or C++ so that they can be run on the WebAssembly virtual machine in the browser. This is interesting as it doesn't require modifying the source code of these programs. Of course, there are limitations, in the kind of underlying APIs and libraries compatible with that platform, and on the sandboxing of the WebAssembly execution engine enforced by the Web browser.

Historically, WebAssembly has been developped so as to allow running games written in C++ for a framework like Unity, in the Web browser.

In some contexts, for instance for tools with an interactive GUI, and processing data retrieved from files, and which don't need very specific interaction with the underlying operating system, it seems possible to port these programs to WebAssembly for running them inside the Web browser.

We have to experiment deeper with this technology to validate its potential for running Virtual Labs in the context of a Web browser.

We used a similar approach in the past in porting a Relational Database course lab to the Web browser, for standalone execution. A real database would run in the minimal SQLite RDBMS, recompiled to JavaScript[11]. Instead of having to download, install and run a VM with a RDBMS, the students would only connect to a Web page, which would load the DBMS in memory, and allow performing the lab SQL queries locally, **disconnected** from any third party server.

In a similar manner, we can think for instance, of a Lab scenario where the Internet packet inspector features of the Wireshark tool would run inside the WebAssembly virtual machine, to allow dissecting provided capture files, without having to install Wireshard, directly into the Web browser.

We expect to publish a report on that last experiment in the future with more details and results.

[11] See the corresponding paragraph `http://www-inf.it-sudparis.eu/PROSE/csedu2015/#standalone-sql-env` in 0

7 Conclusion

The most promising architecture for Virtual Lab deployments seems to be the use of **containers on a PaaS platform** for deploying virtual desktops or virtual application GUIs available in the Web browser.

This would allow the controlled fabrication of Virtual Labs containing the exact bits needed for learners to practice while minimizing the delays.

Still the need for **always-on connectivity can be a problem**.

Also, the potential for inter-networked containers allowing the kind of multiple nodes and collaborative scenarii we described, would require a lot of expertise to develop, and management platforms for the MOOC operators, which aren't yet mature.

We hope to be able to report on our progress in the coming months and years on those aspects.

8 References

[0] Olivier Berger, J Paul Gibson, Claire Lecocq and Christian Bac "Designing a virtual laboratory for a relational database MOOC". International Conference on Computer Supported Education, SCITEPRESS, 23-25 may 2015, Lisbonne, Portugal, 2015, vol. 7, pp. 260-268, ISBN 978-989-758-107-6 - DOI: 10.5220/0005439702600268 (preprint (HTML))

9 Copyright