Open Distribution of Virtual Containers as a Key Framework for Open Educational Resources and STEAM Subjects

Alberto Corbi and Daniel Burgos
Universidad Internacional de La Rioja (UNIR)

Abstract: This paper presents how virtual containers enhance the implementation of STEAM (science, technology, engineering, arts, and math) subjects as Open Educational Resources (OER). The publication initially summarizes the limitations of delivering open rich learning contents and corresponding assignments to students in college level STEAM areas. The role that virtual containers can play in current distant education is then discussed, starting by reviewing related teaching efforts around the use of legacy virtual machines. We then focus on the superseding container technology and how it can bridge the gap between online students, humble computing resources, teachers and IT specificities. As a practical example, we present an experience carried out at the online School of Engineering & Technology at Universidad Internacional de La Rioja (UNIR). Within the context of a subject about Physics for Computing Engineers, we describe the satisfactory evolution from using conventional software distribution methods towards the transition to virtual containers. Thanks to this virtualization approach, the necessary student activities can be implemented, the required software tools can be easily distributed, and the accompanying documentation can be seamlessly presented. The results show how student engagement and satisfaction increased over time, partly because of the easiness introduced by the container technology. Our experience proves that combining containerized educational resources and free and open distribution channels can be one of the cornerstones of a new OER approach in STEAM subjects.

Keywords: virtual containers, STEAM, Open Educational Resources, content distribution platforms

1. Limitations of open educational resources in STEAM

Open Educational Resources (OER) are reaching a relevant level of acceptance in the college-level academic sphere, as the authors in (Zancanaro et al., 2015), (Hatzipanagos and Gregson, 2015) and (Allen and Seaman, 2016) have summarized. Many reputed academic institutions are already distributing free digital content: California State University (http://als.csuprojects.org), MIT OpenCourseWare (http://ocw.mit.edu),Washington State University (https://teach.wsu.edu/oer) and the Tufts Open Courseware (http://ocw.tufts.edu), among others. In addition, non-profit institutions are offering free OER materials like OpenStax (http://cnx.org) and the Open University (http://www.open.ac.uk). Furthermore, there are associations whose goal is to bring together open educational resources and OER creators, contributors, students, and consumers, such as the Open Education Consortium (https://oerconsortium.org), EDUCASE (https://library.educause.edu), OERu (https://oeru.org), OER Foundation (http://wikieducator.org/WikiEducator:OER_Foundation), and OERCommons (https://www.oercommons.org).

As pointed out by (Albright, 2005; Pearce et al., 2012), open educational resources have contributed to the democratization of education by allowing students and teachers to live together in a framework of mutual benefit. Also, as (Burgos, 2006) highlights, the non-hierarchical relationships that emerge between OER learners, OER creators, and teachers contribute very positively to the improvement of formal and informal learning settings, inter-personal interaction, and the overall educational process. Finally, (Downes, 2007) has written a very thorough review regarding sustainable paradigms for open educational resources and discusses several distribution, technical, staffing and funding models.

An OER typically consists of an electronic/multimedia teaching, learning, or even research resource. These resources are mainly available at no cost and under license types that have very few restrictions. Within the contents of an OER, the student can find any material envisioned for educational purposes (i.e., textbooks, related readings, interactive simulations, games, quizzes, assessment software, etc.).

In computing and engineering education, there already exist plenty of open source software initiatives. However, they cannot strictly be considered as OER since usually no companion instructions or teaching guides (i.e., proposed activities, starting tutorials, auto-correction tools, etc.) can be found. For instance, even though they are *de facto* open resources, all open software repositories like Github or Sourceforge and the code hosted inside them, cannot be considered as OER given that they do not necessarily aspire to play an educational role.

Reference this paper as: Corbi A and Burgos D, "Open Distribution of Virtual Containers as a Key Framework for Open Educational Resources and STEAM Subjects" *The Electronic Journal of e-Learning Volume 15 Issue 2 2017*, (pp126-136) available online at www.ejel.org

Besides, traditional educational software deployment in institutions can become a tedious task, which is usually delegated to teachers, students, and other *non-technical* staff. This difficulty arises from the huge variety of computer systems and architectures. In the case of cross-institutional collaboration, the situation worsens, as the authors in (Nerantzi, 2012) examine in detail. The lack of documentation when manipulating these contents also adds extra complexity. OER (and any kind of software-based learning material) not only require access to these digital resources by the student, but they also require knowledge for their installation, configuration and proper use, as summarized in Fig. 1.

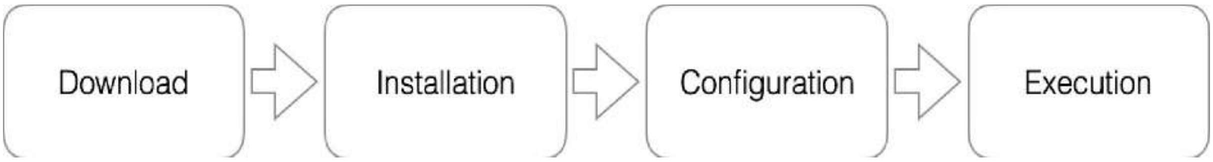


Figure 1: Usual phases in the implementation and use of a software tool.

In face-to-face teaching, the process just described can take place with the means and resources of the educational institution, and the student can ask for help when needed. Nevertheless, in distant learning, the student finds him/herself alone facing all types of technical and learning difficulties. This loneliness can be felt even more intensely in the specific case of OER, where students do not have any institutional or official tutoring contact with the creator of the learning resource and therefore cannot ask for any sort of support.

Some of the required tools entail a huge level of complexity to achieve their proper deployment. In addition, other learning contents require very subtle computing environments such as:

- Specific operating systems and versions.
- Pre-installed libraries, frameworks, and runtimes like Java, Python, dotNet, etc.
- Specific user permissions or admin rights for installing and running software.
- Specific hardware: processor, memory amount, GPU capabilities, etc.

Faced with these situations, the only alternatives are the following:

- Require the student to acquire or replicate the architecture and software conditions necessary for the activities and contents taught.
- Allow the students remote access to a controlled working environment, which is deployed and managed by the institution or by third parties (i.e., cloud-based hostings such as Microsoft Azure, Salesforce Heroku or Redhat Openshift),
- Limit the underlying technologies needed by the learning resource to those that enjoy a broad consensus and level of adoption, i.e., international standards such as W3C and HTML5, ECMA and C#, or ISO and C++.
- Virtualize each working environment through the so-called virtualization technologies, which are tackled in this article.

From our point of view, the only option from the listed above that can be successful for a vast majority of scenarios is, undoubtedly, virtualization, which will be justified below. There is a current tendency to think that cloud-based solutions are the best answer to the above-mentioned problems. Nevertheless, some applications require significant computational resources to be executed in a time-sharing/remote environment and performance may decline significantly in these cloud scenarios. In this paper, we propose virtual containers (discussed in detail in Section 3) as a vehicle for delivering complex educational resources. We study a specific implementation of these containers named Docker.

Docker concentrates a large portion of its appeal in the management of an openly accessible repository of virtualized environments or *images*. This repository behaves like an OER management tool and can be considered to be an OERaaS platform (*OER as a Service*) that is ready for use in distant learning scenarios. This attractive characteristic will be discussed in Section 4, but before addressing this possibility further, we will first discuss the contributions of legacy/classic virtualization technologies in e-learning.

2. State-of-the-art in virtualization of e-learning tools and OERs

In its traditional or *classic* conception, virtualization (also referred to as *legacy virtualization* in this paper), is the implementation of a partial or complete hardware component or system by means of software exclusively. These emulated components can include disks, processors, network infrastructures, peripherals, or full desktop computers (including graphics, media playback, and sound). Virtualized resources are managed by a so-called *hypervisor*, which adds an abstraction layer between the real hardware (*host*) and the virtual scenario (*guest*). The main companies related to classic virtualization are Oracle, Parallels, Citrix, and VMWare. On the other hand, the main open source projects are the well-known VirtualBox and QEMU.

Even when dealing with virtualized environments, the efficiency and versatility achieved by modern commercial and open/non-commercial solutions almost match those achieved by host systems (Soares Boaventura et al., 2014; Seo et al., 2014). Fig. 2 shows an example of the speed that is attainable in database access (VMware, 2008).

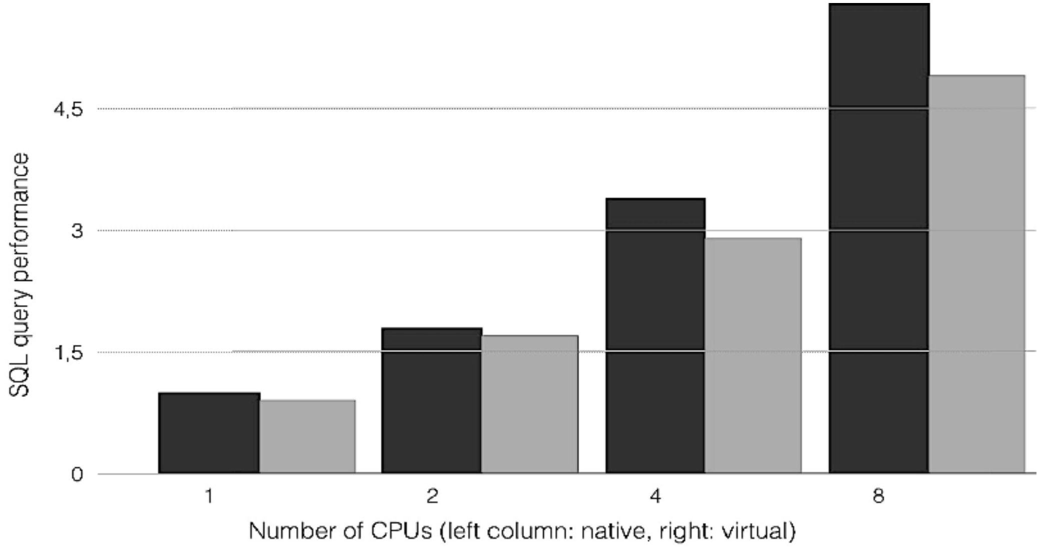


Figure 2: Native vs. virtual performance in SQL access to a database.

This type of virtualization has a huge disadvantage: each time a new activity or (educational) content is created, it has to be wrapped by a new complete virtual machine. This usually entails greater upload and download bandwidth requirements for both the student and the institution. It may also involve a reduction in performance when several exercises, and thus several machines, are executed concurrently. This disadvantage can be overcome thanks to virtual containers, which are discussed in Section 3.

Each new virtual machine is, in turn, distributed following specific conventions and formats (OVF, VDI, VDMK, etc.) agreed upon reputed companies and projects that are affiliated with the Open Grid Forum (OCCI-WG, 2010).

There is an interesting study carried out by (Bruce, 2010) about the application of virtual machines in education that analyzes the barriers that are preventing the entry of virtualization in schools. The main barriers are the following:

- Lack of skills and knowledge by teachers and students.
- Lack of resources by educational centers and institutions.
- Disagreements between the stakeholders in the educational system.

However, IBM (IBM, 2007) predicts an unstoppable rise of virtualization technology in all areas and recommends IT services in schools to join this trend. Other research groups in educational technologies such as (Nauczycielski, 2011) have performed a comprehensive analysis of existing virtualization technology and its application to the teaching of subjects related to networking.

Traditional virtualization and virtual machines have been used as educational tools in STEAM since their technological birth. Typically, these educational resources emulate full desktop environments with everything necessary for the student to begin solving the required tasks with the greatest ease of use possible. Usually, the choice of virtual machines has mainly been influenced by the degree of complexity of the resource to be used by the student. If a resource required an arduous configuration for its use, then, a properly configured virtual machine seemed to be the best choice. For example, that is the case for GATE (*Geant4 Application for Tomographic Emission*). GATE offers the download of a traditional virtual machine based on the popular Ubuntu Linux desktop distribution with all the software necessary to directly operate with this framework related to particle and medical physics.

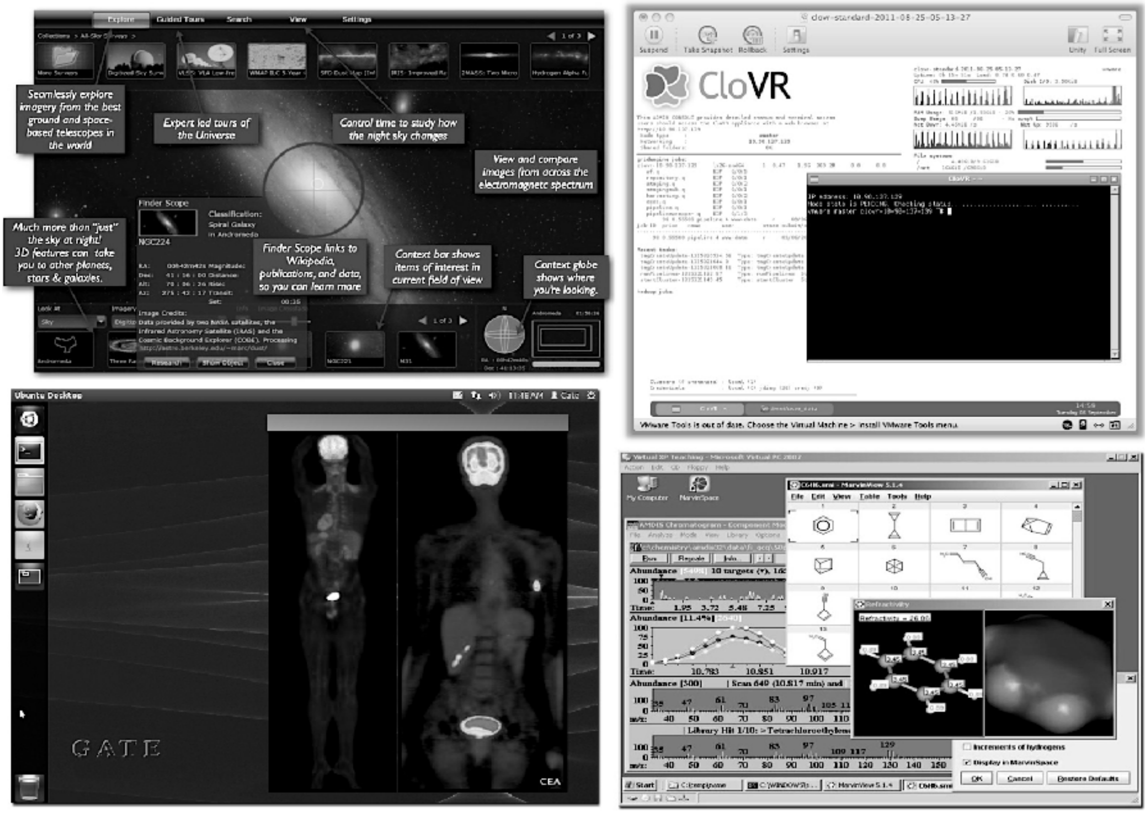


Figure 3: Some educational resources that are distributed as classic virtual machines (WWT, CloVR, vGATE, and a VirtualBox image for chemistry teaching).

Similarly, the authors in (Goodman et al., 2012) have developed a virtual machine-based learning environment for astronomy (WWT). The CloVR project (Angiuoli et al., 2011) shares the same goals, but it is focused on teaching genetics. The authors in (Kind et al., 2009) have implemented a VirtualBox virtual machine for teaching chemistry, and researchers in (Hamada, 2009) have done something similar in math-related subjects. The BioImg project (Dahlö et al., 2015) is aimed at centralizing a complete repository of legacy virtual machines with learning resources for the teaching of biology. The web pages of these projects are shown in Table 1. Fig. 3 shows some screenshots of these learning environments.

Table 1: Some STEAM educational environments that use classic or traditional virtual machines.

Project	URL	Description
GATE	opengatecollaboration.org	Simulations in medical physics and radiotherapy
WWT	worldwidetelescope.org	Framework for working with a virtual telescope
CloVR	clovr.org	Genetic analysis
BioImg	bioimg.org	Exercises in bioinformatics

3. Virtual containers

Virtual containers (Rosen, 2014) can be considered light virtual machines that are typically based on a shared GNU/Linux system. They are designed to run an instance of a specific application (and not a *canonical* full-screen desktop environment with a complete set of applications). A container's mission is usually to implement a web service: a Ruby on Rails, NodeJS, or PHP application that owns an interactive TCP/IP port. The way to do this is by running a virtual machine that implements just the components that are strictly necessary for such a service to run.

Containers have become the great allies of programmers, system administrators, and DevOps (*development and operations*) because they can be easily deployed on any computer infrastructure that has the minimum support pre-installed. The main advantage is its lightness and the ability to work in both development and production environments. The difference from traditional virtual machines is that all contained (or *containerized*) applications share the same underlying software layer, as shown in Fig. 4.

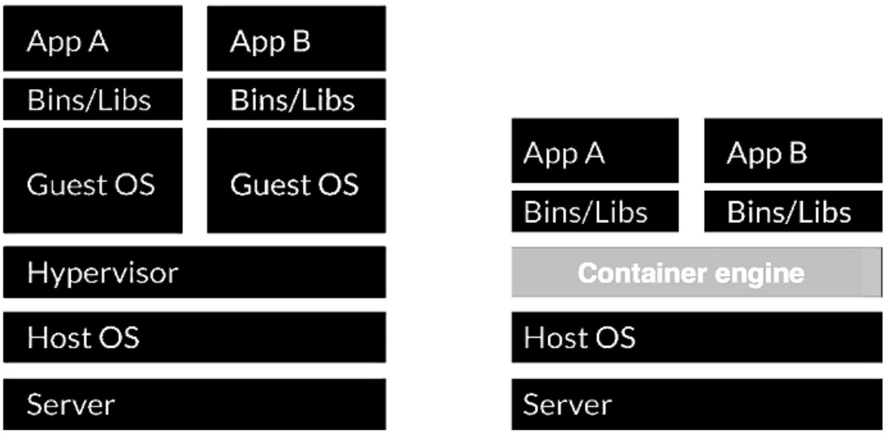
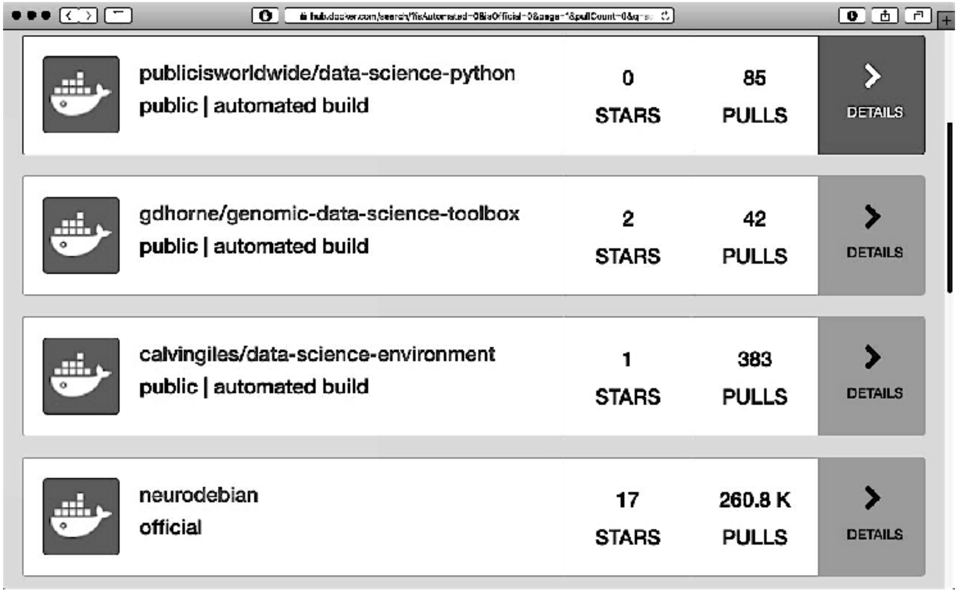


Figure 4: Classic virtualization (left) vs. container virtualization (right).

The main projects that use virtual containers are Xen (Barham et al., 2003), LXC (Rosen, 2014), Docker (Liu and Zhao, 2014), KVM (Kivity et al., 2007), OpenVZ (Kolyshkin, 2006), VMware ESX (Muller and Wilson, 2005), and libvirt (Bolte et al., 2010). There are also interesting comparisons between these technologies such as those carried out by (Deshane et al., 2008; Che et al., 2010, and Fragni et al., 2010).

In recent years, virtual containers have occupied an important niche in systems administration (Rosen, 2014). Container technology is currently considered to be the best answer to the problem of *how to get software to run reliably when shifted from one computing environment to another*. A container consists of a complete and packaged runtime bundle, which includes the target application and all its dependencies (i.e., linked or static libraries, helper programs, state or configuration files, etc.). By *containerizing* an application and its dependencies, differences in the underlying infrastructure are abstracted away. In contrast with conventional virtualization technology such as VMware Fusion, Parallels Desktop, Oracle VirtualBox, etc., several containerized applications share a single operating system (OS) kernel. This makes them lighter and less resource-hungry than conventional virtual machines (less than 100 megabytes or even less) and enables the distribution of large scale educational environments. As an immediate consequence, a container can easily be run either on humble local user/student hardware or on less expensive commercial cloud infrastructures (Joy, 2015).

Many projects related to the core technologies in containerization have emerged recently and many computer engineering companies, communities, and associations (both large and small) are involved. KVM from Open Virtualization Alliance, ESX from VMWare, or Docker from dotCloud are just a few examples (Che et al., 2010). As with other alternatives, Docker implements a simple, high-level interface to provide lightweight virtual environments that run isolated processes. However, Docker has a key advantage over other choices, the so-called *Hub*. The Docker Hub is a free online registry service for distributing containers (Fig. 5). It also provides search utilities for container discovery, management, and team collaboration (Hagstrom and Essary, 2009). As we suggest in Section 4, the Docker Hub can implicitly behave as a service for OER distribution and may be regarded as an OERs as a service (OERaaS) environment.



publicisworldwide/data-science-python
public | automated build
0 STARS 85 PULLS

gdhorne/genomic-data-science-toolbox
public | automated build
2 STARS 42 PULLS

calvingiles/data-science-environment
public | automated build
1 STARS 383 PULLS

neurodebian
official
17 STARS 260.8 K PULLS

Figure 5: The web site of the Docker Hub.

Undoubtedly, the main application of virtual containers is the distribution of services and applications. However, containers are also catching the attention of science research groups as a means of assuring the reproducibility of experimental results. For example, the authors of (Boettiger, 2015) examined this possibility in the case of Docker and (Clark et al., 2004) for Xen. However, its application has hardly been explored in the academic realm, or mode specifically, as a core teaching resource in distant education. Therefore, we propose the suitability of Docker containers to ensure a correct, simple, uniform and open distribution of educational content. More specifically, we suggest the use of Docker and its Hub for the distribution of open learning resources. Another important reason for selecting Docker is its strong open source foundation and its healthy developer community only comparable to other worldwide relevant projects such as the Linux kernel.

4. The Docker HUB as an OER distribution platform

One of the most successful implementations in the ecosystem of virtual containers is Docker (Tuomas, 2015). This virtual container alternative works with the concept of inter-connectable and inter-dependent images, which gives it great flexibility and explains the commercial success that it is already having in its thus far short life. These images are container snapshots that fit together like puzzle pieces and form a virtual operating environment. Each container incorporates just the frameworks (libraries, binaries, configuration files, support scripts, etc.), specific configuration files, and the software necessary to perform one task.

Docker has a public, open, and free container repository that handles thousands of these snapshots that are already ready to be downloaded and deployed. This repository is called the Docker Hub (Fig. 5) or simply *Hub*. Because registration is free and does not involve any royalty, a registered user can upload images to the Hub and share them with a vast and growing community of users. Some of these images have markedly academic objectives because they are eminently designed to recreate specific educational environments for many of the knowledge areas of STEAM subjects. Therefore, the Hub behaves as a *de facto* OERaaS platform from which hundreds of educational resources are distributed and served daily.

Table 2 shows some Docker images that are related to science and education and in which repository (within the Hub) they can be found. A repository name usually has two parts: author/image. The first one refers to the author of the image and the second one to the image itself.

Table 2: Some repositories of container images related to education in the Docker Hub.

Hub repository	Description
bwawrik/bioinformatics	Bioinformatics through the Python programming language
arkadi/mathics	Alternative to Mathematica from Wolfram Research
sagemath/sagemath	Environment for symbolic and numerical mathematics
official/scratch	Environment from MIT to teach programming to kids
official/gazebo	Interactive robot simulation

The images belonging to the *official* repository are considered to be of great relevance in specific environments and are usually developed, maintained, and uploaded by institutions with a long tradition. This is case of the famous Scratch software (Resnick et al., 2009) from the MIT, in the field of education.

5. Research

As a practical example, we discuss the experiences carried out at the online School of Engineering & Technology at Universidad Internacional de La Rioja (UNIR). In the context of a Physics for Computing Engineers subject, we describe the satisfactory evolution from using conventional software distribution methods towards the transition to Docker containers for broadcasting each assignment's underlying software. Each homework set was given to the student as a virtual container. Our working hypothesis is that virtual containers can significantly improve the student experience when having to solve complex mathematics, physics and other STEAM-related college level exercises.

These assignments ranged from simple physics problems to be solved with a set of Python scripts to more complex simulation scenarios that required intricate software outlines. This is the case of an exercise related to the study of sub-atomic particle and photon tracks/collisions calculated with the legendary Geant4 package from the CERN. All of the containers deployed the necessary software tools for each physics task, examples, and companion resolution guides, thus freeing the learner from these duties and allowing him/her to concentrate on the problem itself and how to solve it. All of the containers (and all of the proposed activities) shared common resources such as the Kernel, basic libraries, or a Python environment.

The above-mentioned taught physics subject (our evaluation scenario) has an eminently applied focus. The methodology followed is the study of the main computing tools for projects that are currently part of modern physics experiments. The commitment to the implementation of these activities is voluntary since they usually involve more time and dedication by the student. That is why this block of exercises (Table 3) is called *alternative activities*. Although they are given some weight in the course grade, their execution and resolution is not mandatory. In early editions of this physics course, each of these tools was distributed in a more traditional way, i.e., through discrete software packages for each operative system that each student had to install on his/her own computer. Despite the careful preparation of each tool, compatibility problems and configuration issues arose in a fairly high number of cases. For this reason, in subsequent academic years, the use of classic virtual machines (complete desktop environments) was favored for certain activities. One of these activities was the particle physics lab. To minimize the complexity related to the deployment and execution of this exercise, a *headless* virtual machine (not desktop-based and without graphical interface) was created. This machine was downloaded by students, who could access it through a SSH session. This session enabled the execution of the necessary calculations. The only drawback related to this way of distributing a learning content of this type is that if this particle physics assignment ever requires (even minor) modifications, a new and complete virtual image has to be rebuilt from scratch.

As with the simulation of fundamental particle interactions, the vast majority of these cutting-edge scientific projects (particle physics, accelerator physics, nuclear medicine, electromagnetism, optics, circuit analysis, etc.) require very specific computing environments that are very difficult to reproduce outside of the research/academic field in which they were conceived. This means that when these tools have to be deployed in a foreign educational environment, technical difficulties may normally arise.

For the 2015-2016 academic year, the School of Engineering decided to move the implementation of some of these activities to Docker-based virtual containers. This has led to huge workflow and methodological simplification for all students since they were only required to install the basic Docker toolchain. Once installed, the students were able to download these resources from the Hub website or through the more modern tool called Kitematic (Fig. 6), which was developed by the Docker team to handle virtual containers in a more convenient way.

Our evaluation process consisted on measuring the rate of satisfaction of students relative to the use of virtual containers and the number of successfully completed tasks with and without virtual container-based technology. These assignments were submitted through the assignments tool in a Sakai-based online campus. Students also filled a simple satisfaction questionnaire at the end of the semester, rating their satisfaction regarding the use of Docker and virtual containers as an appropriate method for designing and delivering remote STEAM labs.

Table 3: Proposed activities executed with virtual containers

Activity	Software	URL
Circuit analysis	NGSpice	http://ngspice.sf.net
Symbolic maths	Maxima	http://maxima.sf.net
Function plotting and charting	GNUPlot	http://gnuplot.info
Optics	GNU Octave	http://octave.sf.net
	OpenCV	http://opencv.org
	Python	http://python.org
Particle physics	Geant4	http://geant4.web.cern.ch
	Root	http://root.cern.ch
Quantum physics	Ruby	https://www.ruby-lang.org
	Java	http://java.com
Word processing of scientific documents	$\LaTeX$	https://www.latex-project.org
	HTML5	https://www.w3.org
Medical physics	DCMTK	http://dicom.offis.de
	ITK	http://www.itk.org
	VTK	http://www.vtk.org
	C++	https://isocpp.org

These images were publicly available on the Docker Hub and they could be downloaded and used by any student, regardless of their institution. For this reason, these images can be considered as OER, and the Docker Hub has played the role of an OERaaS.

ıdo
r de ejemplo

eléctricos
n de circuitos con
geda-gaf

n texto científico
n de textos técnicos
istribución completa

Física con Geant4
Física de partículas con el conocido entorno de simulación del CERN
♡1 ⇩5

Física médica
Visualización y análisis de imágenes DICOM con DCMTK
♡1 ⇩5

Física cuántica
Introducción a la física cuántica de la mano del framework ROOT del CERN
♡1 ⇩5

Óptica geométrica
Calibración geométrica de cámaras mediante el paqu de cálculo Octave
♡1 ⇩5

Efecto fotoeléctrico
Cálculo de la constante de Planck con Ruby
♡1 ⇩5

Funciones y cálculo
Contiene todo lo necesario para matemáticas básicas GNUPlot y Maxima
♡1 ⇩5

Short Description
IPython GEANT4 with a basic ex python lectures included

Full Description
Run this by calling:
```
docker run -p 8888:8888
```
Then type either `localhost:8` `192.168.59.103:8888` (win chrome or firefox browser.
An example of what this docker
More detailed install instructions

Figure 6: Kitematic allows the free download of container images created for the physics subject presented in this research

6. Results and discussion

Our results show how student engagement and satisfaction increased over time, partly because of the ease and swiftness introduced by the container technology in the distribution of each of the physics lessons mentioned above (Fig. 7).

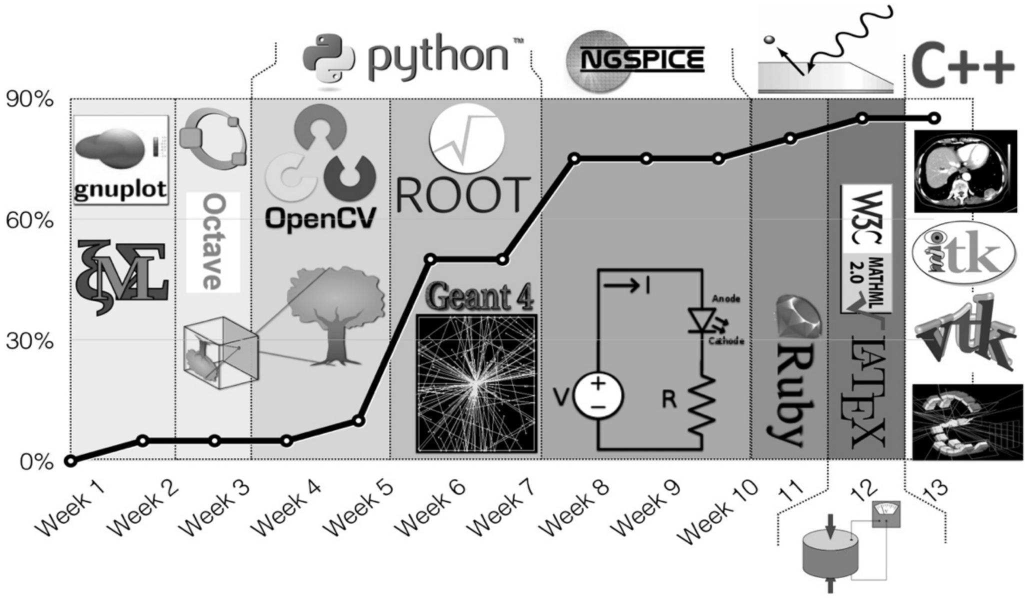


Figure 7: Evolution in the percentage of students (superimposed black line) committed to execution of alternative activities based on virtual containers. In each week, a complete new technology and science problem (entailing a radically different computing scenario) was introduced. Some of these technologies and frameworks are summarized in Table 3.

These resources were openly available in the Docker Hub for one semester. Enrolled students accessed, downloaded, and run them as plain OER in their own personal host systems. Our experience proves that combining containerized educational resources and free and open distribution channels can be one of the cornerstones of the OER approach in STEAM subjects. Fig. 8 shows the evolution in the commitment of students to the elaboration and submission of alternative activities involving richer computing scenarios. Clearly, the 2015-2016 semester represents a huge difference (in student commitment) when compared against the previous academic years.

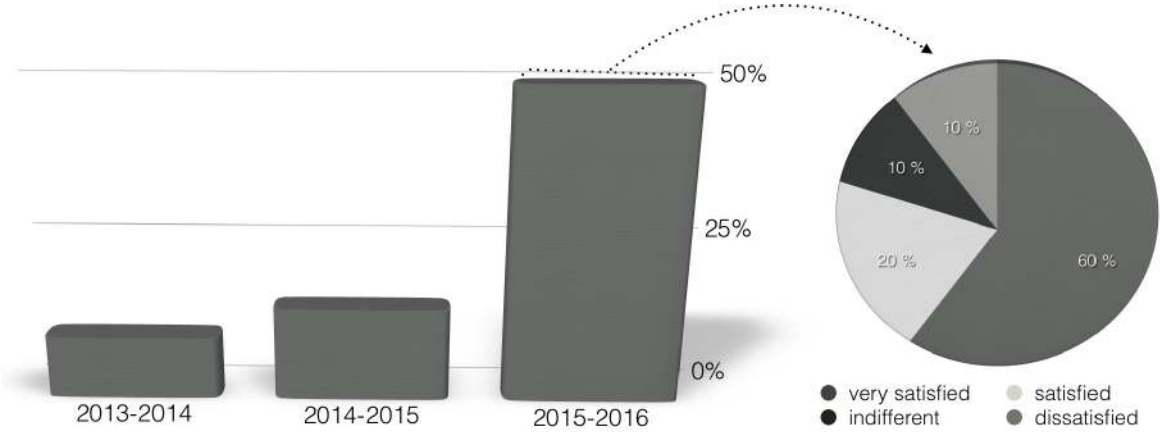


Figure 8: Evolution in the commitment and dedication to alternative activities based on the use of scientific software tools. Only in the 2015-2016 semester, virtual containers were used as part of the teaching methodology. The vertical axis entails the percentage of submitted alternative activities (discussed in Section 5). The pie graph shows the degree of student satisfaction with the technology of virtual containers (applicable only in the 2015-2016 academic year, when the satisfaction questionnaire was handed in to the students).

Similarly, during the 2015-2016 academic year, interest has been increasing with a significant rise in week 6 (Fig. 7). During that week, the students carried out the same exercise related to particle physics (described above). In this activity, the students simulated a beam of particles and their possible interactions with matter and detectors. The ease of implementation of this task by means of a virtual container attracted a large number of students to continue solving the rest of the proposed activities, which were also distributed as lightweight interdependent containers.

Also, our results correlate with the increasing number of Docker images available in the Docker Hub and that are related to education in STEAM subjects (Fig. 9).

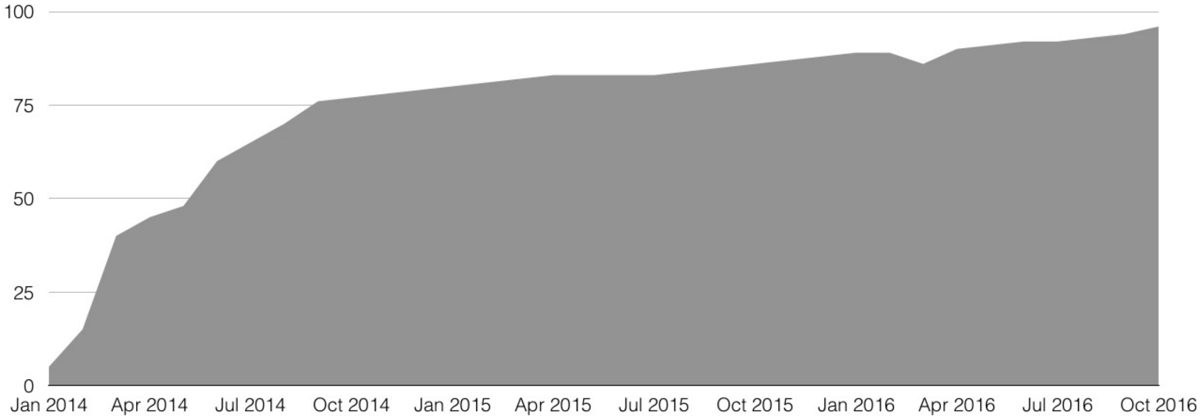


Figure 9: Evolution in time on the number of Docker images dedicated to STEAM education.

7. Conclusions

The simplification of the distribution of computing environments in education is a key element in attracting students to the use of modern and highly complex STEAM learning tools. The virtual containers represent a powerful tool for distribution of OERs. In this article, we have focused on the Docker project and its Hub platform, which are aimed at the easy and open delivery of virtual containers. We have demonstrated through an actual case study how this tool can operate as an OERaaS platform. Throughout the duration of this case study, we perceived and measured a progressive increase in the interest and commitment of students towards the use of the proposed educational tools. As a future line of work, our research group is considering the use of *unikernels* as a method for delivering rich technological and scientific content (including related assignments). Unikernels represent and deeper simplification of the virtual container approach, given that all necessary computing elements (operative system kernel, basic libraries, frameworks, drivers, scientific application, etc.) reside in just one minimal, binary, executable file. The main advantages of unikernels over containers are the improved security, the small footprint and the increase in speed.

Acknowledgements

This research is partially funded by UNIR Research (http://research.unir.net) and the Research Institute for Innovation & Technology in Education (UNIR iTED, http://ited.unir.net), at Universidad Internacional de La Rioja (UNIR).

References

Albright, P. (2005). Open educational resources open content for higher education. In *Internet Discussions Forums*.

Allen, E. and Seaman, J. (2016). Opening the Textbook: Educational Resources in U.S. Higher Education, 2015-16. Babson Survey Research Group.

Angiuoli, S. V., Matalka, M., Gussman, A., Galens, K., Vangala, M., Riley, D. R., Arze, C., White, J. R., White, O., and Fricke, W. F. (2011). CloVR: a virtual machine for automated and portable sequence analysis from the desktop using cloud computing. *BMC bioinformatics*.

Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., and Warfield, A. (2003). Xen and the art of virtualization. *ACM SIGOPS Operating Systems Review*.

Boettiger, C. (2015). An introduction to Docker for reproducible research. *ACM SIGOPS Operating Systems Review*.

Bolte, M., Sievers, M., Birkenheuer, G., Niehörster, O., and Brinkmann, A. (2010). Non-intrusive virtualization management using libvirt. In *Proceedings of the Conference on Design, Automation and Test in Europe*.

Bruce, J. (2010). Desktop virtualization in K-12 schools. Technical report, Center for Digital Education.

Burgos, D. (2006). The structure and behavior of virtual communities engaged in informal learning about e-learning standards. PhD Thesis, European University of Madrid.

Che, J., Yu, Y., Shi, C., and Lin, W. (2010). A synthetical performance evaluation of OpenVZ, Xen and KVM. In *Services Computing Conference (APSCC), 2010 IEEE Asia-Pacific*.

Clark, B., Deshane, T., Dow, E. M., Evanchik, S., Finlayson, M., Herne, J., and Matthews, J. N. (2004). Xen and the art of repeated research. In *USENIX Annual Technical Conference*.

Dahlö, M., Haziza, F., Kallio, A., Korpelainen, E., Bongcam-Rudloff, E., and Spjuth, O. (2015). Bioimg.org: A catalog of virtual machine images for the life sciences. *Bioinformatics and Biology Insights*.

Deshane, T., Shepherd, Z., Matthews, J., Ben-Yehuda, M., Shah, A., and Rao, B. (2008). Quantitative comparison of xen and kvm. *Xen Summit, Boston, MA, USA*.

Downes, S. (2007). Models for Sustainable Open Educational Resources, National Research Council of Canada.

Fragni, C., Moreira, M. D., Mattos, D. M., Costa, L. H. M., and Duarte, O. C. M. (2010). Evaluating Xen, VMware, and OpenVZ virtualization platforms for network virtualization. *Universidade Federal Rio de Janeiro*.

Goodman, A., Fay, J., Muench, A., Pepe, A., Udomprasert, P., and Wong, C. (2012). Worldwide telescope in research and education. *arXiv preprint arXiv:1201.1285*.

Hagstrom, A. and Essary, B. (2009). Managing team software projects using virtual containers. US Patent 7,631,006.

Hamada, T. (2009). Knoppix/math: A live system for enjoying mathematics with computer. *ACM Commun. Comput. Algebra*.

Hatzipanagos, S. and Gregson, J. (2015). The Role of Open Access and Open Educational Resources: a Distance Learning Perspective University of London Centre for Distance Education , Visiting Fellow , UK. 13(2):97–105.

IBM (2007). Virtualization in education. Technical report, IBM Global Education.

Joy, A. M. (2015). Performance comparison between linux containers and virtual machines. In *International Conference on Advances in Computer Engineering and Applications*.

Kind, T., Leamy, T., Leary, J. A., and Fiehn, O. (2009). Software platform virtualization in chemistry research and university teaching. *Journal of Cheminformatics*.

Kivity, A., Kamay, Y., Laor, D., Lublin, U., and Liguori, A. (2007). KVM: the Linux virtual machine monitor. In *Proceedings of the Linux symposium*.

Kolyshkin, K. (2006). Virtualization in linux. *White paper, OpenVZ*.

Liu, D. and Zhao, L. (2014). The research and implementation of cloud computing platform based on docker. In *11th International Computer Conference on Wavelet Active Media Technology and Information Processing*.

Muller, A. and Wilson, S. (2005). Virtualization with VMware ESX server.

Nauczycielski, W. (2011). Desktop virtualization as a modern solution in education. In *Ročník Národnej Konferencie*.

Nerantzi, C. (2012). A case of problem based learning for cross-institutional collaboration. *Electronic Journal of e-Learning*, 10(3):306–314.

OCCI-WG (2010). Open cloud computing interface-infrastructure models. Technical report, Open Grid Forum.

Pearce, N., Weller, M., Scanlon, E., and Kinsley, S. (2012). Digital scholarship considered: How new technologies could transform academic work. *in education*, 16(1).

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., et al. (2009). Scratch: programming for all. *Communications of the ACM*, 52(11):60–67.

Rosen, R. (2014). Linux containers and the future cloud. *Linux Journal*.

Seo, K.-T., Hwang, H., Moon, I., Kwon, O., and Kim, B. (2014). Performance comparison analysis of linux container and virtual machine for building cloud. *Advanced Science and Technology Letters*.

Soares Boaventura, R., Yamanaka, K., and Prado Oliveira, G. (2014). Performance analysis of algorithms for virtualized environments on cloud computing. *IEEE Latin America Transactions*.

Tuomas, V. (2015). *Advantages of Docker*. PhD thesis, University of Jyväskylä.

VMware (2008). SQL Server workload consolidation. Technical report, VMware.

Zancanaro, A., Todesco, J. L., and Ramos, F. (2015). A bibliometric mapping of open educational resources. *The International Review of Research in Open and Distributed Learning*, 16(1).