Teaching parallel computing with container virtualization

Joshua Higgins*, Violeta Holmes
High Performance Computing Research Group,
University of Huddersfield
UK
Email: *joshua.higgins@hud.ac.uk, v.holmes@hud.ac.uk

***Abstract*—Incorporating modules that equip students with parallel programming and high performance computing skills into core computing and engineering courses poses unique technical challenges. A typical PC laboratory environment may not be suitable, and allowing learners access to a production resource may introduce an unacceptable risk, if it is even possible. For decades, Beowulf clusters have offered an attractive solution where a system can be built using commodity components, often by the students themselves. This can instil in the learner both an understanding of parallel code and how it is orchestrated on the hardware - a strategy used successfully in delivering courses at the University of Huddersfield. However, there are two problems; Firstly, the depth of knowledge required to be learnt in order to successfully build, program and profile a cluster. Secondly, the use of seemingly antiquated tools in order to achieve this. In this paper, we outline some of the experience of integrating parallel and distributed computing into our courses. In addition, we present a novel, cross-platform virtual cluster toolkit developed to address the technical requirements of delivering such courses.**

I. Introduction

High Performance Computing (HPC) drives scientific innovation in academia and industry. It is no longer the domain of theoretical scientists or software engineers. It is a tool that allows fast answers to research problems, such as through simulation of physical phenomena, fluid dynamics and molecular interactions. However, a new user wishing to utilise these tools holds a significant knowledge defecit.

In a report to the UK Government in 2012, the E-infrastructure Leadership Council identified a need for training researchers with the advanced skills required to take advantage of HPC systems[1]. It recommends that the foundation for this training must be laid at the undergraduate level. There are some undergraduate courses in the UK that offer these skills, but they are few and far between. Typically, they focus on analytics, machine learning, visualisation and multi core programming. Until recently, only 2 universities offered an undergraduate course specifically in HPC[2].

However, the demand for HPC capabilities, especially from early career researchers, will continue to grow. This highlights the need for effective methodologies in order to train users to take advantage of these capabilities. The solutions we have used in the past suffer from being unmaintained and out of date, too opinionated and inflexible. In addition, HPC is beginning to encompass more than just Message Passing (MPI) code - whether we can teach it or not depends somewhat on whether the resource can adapt or not.

In this paper, we suggest that virtualization offers a potential solution to these concerns, and present a tool developed based on our experience of integrating parallel and distributed computing topics into our courses.

The rest of this paper is organized as follows: Section II contains a brief introduction to container virtualisation, Section III introduces the Parallel Computer Architecture modules at the University of Huddersfield, Section IV outlines the problems that we have identified with the current teaching method, Section V details the implementation of the new tool in the classroom. Finally, the usability of the tool is evaluated in Section VI.

II. Container Virtualization

Typically, virtualization employs a *hypervisor* that is responsible for providing the virtual interfaces that resemble a real system - the *Virtual Machine* (VM) - to the guest Operating System (OS)[3]. This allows multiple instances of arbitrary Operating Systems to be running on a single host. There are inherent overheads in this process that can impact the performance and increase the resource consumption of a machine. Because of this, they are not typically employed in HPC systems as a mode of execution.

However, container virtualization offers a lightweight alternative which relies on the host OS's kernel to provide isolation between guests rather than requiring a layer of hardware abstraction[4]. It is conceptually similar to a `chroot`, but with greater isolation. There are many implementations of container runtimes, such as OpenVZ[5], LXC[6], Docker[7] and Singularity[8]. The container runtime provides a tool to manage the lifecycle of the containers on one or more systems. Docker has gained widespread acceptance, with a focus on standardization of the container format and a workflow that promotes container sharing and reuse.

Performance benchmarking has shown that application performance can equal the performance of running without virtualization, especially in a parallel execution context[9][10]. In this way, containers have become popular in the HPC community for packaging applications.

III. Parallel Computer Architectures at the University of Huddersfield

The University of Huddersfield offers a series of *Parallel Computer Architecture* modules to both undergraduate and postgraduate students, available as a core module and an option, depending on course pathway[11].

The undergraduate module, focusing on cluster and grid computing, is taught over a period of 22 weeks, with 2.5 hours of contact time per week. The postgraduate module additionally includes an introduction to cloud technologies, taught over a period of 12 weeks with 4 hours of contact time per week. The time available to students on both is comparable; Masters students are expected to attain competency in the respective topics faster and to a higher degree than the Undergraduate students. This is reflected in the design of the lecture and laboratory sessions.

A. Learning Outcomes

The learning outcomes of the module aim to provide students with a broad introduction to the fundamental aspects of parallel and distributed processing:

- Understand the need for and evolution of computer clusters and grids in the context of processor- and data-intensive applications.
- Gain insight in the fundamental components of Cluster environments, such as commodity components for clusters, network Services and communication, cluster middleware, resource management and programming environments
- Gain insight in the fundamental components of grid environments, such as authentication, authorization and resource discovery
- Understand scalability issues by using performance metrics and measures, to quantify speed up in parallel processing applications

B. Delivery and Assessment

The relevant concepts, algorithms and programming models are presented through lectures, followed by group work during the laboratory based sessions. This allows the development of understanding and enhancement of practical skills, whilst providing a mechanism for continuous formative assessment. During the course of the term, the students will build, program and profile a computer cluster. They will learn how to install and configure the operating system, middleware and toolchain in order to compile and run an MPI code. The postgraduate students additionally gain experience with the cloud through deployment of a web server and load balancer on a public cloud provider. The undergraduate students continue the cluster project into the second term, implementing grid technologies for authentication and authorization in order to share their cluster resources with other groups in the same classroom. Everything required is provided in the laboratory sessions: hardware is made available from the University's rolling replacement program and software is hosted on an internal repository. Support is provided by acamedic staff in the laboratory sessions for all aspects of the module.

The principle strategy for assessment is through project based learning. Each group is tasked with benchmarking the constructed cluster using a suitable method. It is usually clear to students, through modifying code to include simple measures such as *wall time* and *ping-pong* latency, where the inefficiencies in their system lies. They must use this information to develop a critical evaluation of the performance of the designed system.

In addition to project work, the final element of assessment is examination.

C. Student profile

The module is a popular choice and attracts students with a broad range of skill levels and proficiency, from both computing and engineering backgrounds as shown in Figure 1. Some of these courses include:

- Computer science
- Software engineering
- Electronic and communication systems

Therefore, to improve accessibility there is no mandatory knowledge or experience that is required in order to join the module. However, regardless of background, most students will have already been exposed to programming, especially in the C language, which is a significant advantage.

Since the introduction of this module, the student numbers have increased steadily year on year. The student feedback is consistently good and it is clear that students are enthused by the practical based problem solving demanded by the module in order to develop confidence in their independent research and learning. Traditionally, at least one student from each cohort completing the module is recruited as a Master by Research or PhD student in the High Performance Computing Research Group.

IV. Limitations of delivery method

Although the *Parallel Computer Architecture* modules have been delivered successfully for several years, a number of issues with this teaching method have become apparent and must be addressed.

A. Hardware sustainability

The hardware used in the delivery of the course is *second hand*, and as such, students must deal with any faults or reliability problems that arise. These faults vary widely, however the most common cause of problems is Random Access Memory (RAM) - modules that have been unseated and not seated correctly, or showing faults on an extended memory test. The incidence of power supply and hard drive failures is also high. This presents both an advantage and a disadvantage. The students gain valuable hands on experience of diagnosing, disassembling and reassembling components that is often not present in other modules. However, when the incidence of problems is so high that it prevents learners from progressing relative to the objectives, there is a problem.

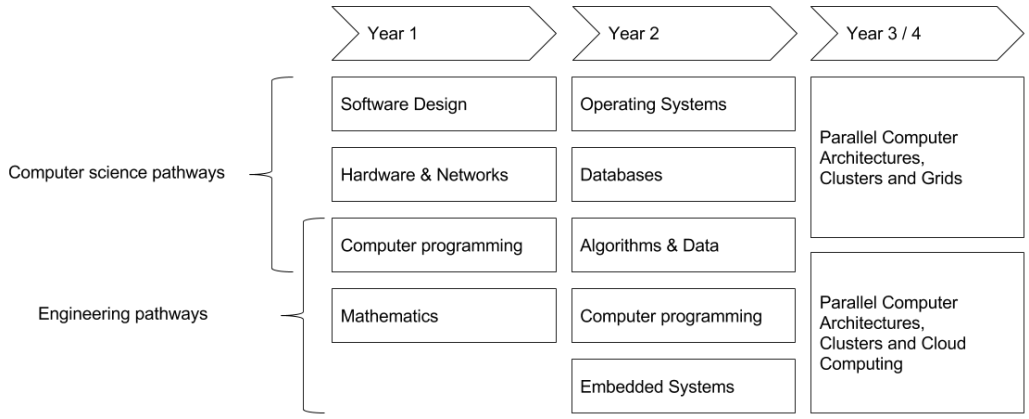

Fig. 1. Incoming students skills profile

Furthermore, faults that occur week on week can result in *two step forwards, one step back* which is frustrating for the learner.

It is difficult to mitigate these problems because it is not sustainable to acquire new hardware for every cohort. In addition, there is no guarantee that the university will continue to make equipment that has been the subject of replacement available for use on the module. Finally, eliminating this problem entirely would be removing a valuable element of the learning experience from the module.

B. Software sustainability

Middlewares that we have used for teaching parallel and cluster computing, such as OSCAR[12], are easy to use which makes them an excellent choice for beginners. However, the OSCAR project is seemingly unmaintained and has not been updated in a number of years[13]. Firstly, it is becoming increasingly difficult to justify it's use where there is a lack of any community activity. This means there are few resources for finding solutions when problems are encountered, and the long term availability of the project's repositories, installation files and documentation is not certain.

Secondly, it requires old versions of the operating system with potentially poor compatibility on newer hardware. This introduces a whole new array of problems and a choice where both options are undesirable: use old hardware with low reliability (more hardware problems), or use new hardware with low compatibility (more software problems).

Other middlewares that are more actively maintained, such as ROCKS[14], are not deemed suitable because of the assumptions made by the developers in how the cluster will be configured. This would require an unreasonable amount of supporting infrastructure to be implemented in the laboratory sessions, reducing the time available to the subsequent activities in programming and profiling the performance. For example, intimate knowledge and *by hand* configuration of the network routing between private and public networks, node boot sequence and node image customization is not appropriate for the skill level of the module, nor within the scope of the learning outcomes.

C. Adaptability

HPC already encompasses a wide range of technologies that do not conform to the typical batch scheduling cluster paradigm. For example, Apache Hadoop[15] is framework used for the storage and processing of big data. Outside of scientific applications, distributed applications are also commonplace, such as clusters of databases and web servers.

When considering these types of software, a difference in understanding of what the term *cluster* means becomes apparent: a system built using a middleware to support one type of cluster does not easily lend itself to supporting another.

The existing solutions cannot quickly adapt in order to meet the requirements of delivering courses that train students with the skills to use these technologies. In past experience, we have found that it is too time consuming to deliver a mixed mode course that covers more than one of these applications, as the majority of time is spent by learners reconfiguring the system rather than learning how to use it effectively.

D. User ability

The assessment of the module's learning outcomes is project based. Building the cluster comprises part of this project, which involves a number of dependent steps that must be completed successfully before moving on to the next. If a student is not able to complete one of the tasks along this critical path, they are unable to complete the work in it's entirety. The experience of configuring the underlying system is valuble, helping to instill in the learner an acute understanding of the relationship between the application and the hardware it is running on. However, the skills and knowledge gained through using the tools are of equal, if not greater, importance. The inability to perform one of the tasks physically inhibits the learner to attempt another because they have not built a working resource that the subsequent tasks depend on.

V. Virtual Container Cluster

The Virtual Container Cluster (VCC) is designed as a toolkit that can be used to build virtualized cluster software environments using Docker containers[16]. A container built

with VCC provides a turn key installation of a cluster middleware, facilitated by a resource discovery engine that performs contextual configuration (such as network addresses, host files, mounts) at run time. This allows a multi-node virtual cluster to be spun up across one or more physical machines quickly, without manual configuration of the individual cluster services if desired, significantly lowering the barrier of entry to setting up the environment.

The memory overhead of each virtual node is low (exactly equal to the memory consumption of each process running in that node). Therefore, it is possible to stage very large environments on a single machine with modest specifications. Whilst it is clear this offers no performance advantage, it provides a useful function as the execution workflow and interactions between processing elements within the system are modelled exactly as they would be in a real system.

The layered filesystem approach enables optimization of the storage space required by reusing common layers between containers. This allows multiple instances to be launched without having to duplicate large filesystem images for each instance.

Regardless of the environment within the container, the version of the host OS is largely irrelevant as long as it supports the Docker runtime. Thus, this methodology takes the path of utilizing new hardware and addressing the resulting software challenges. In addition, it enables cross-platform support as the Docker runtime is available on Windows, Mac and Linux based operating systems. This opens up the opportunities for the training environment to be hosted in unmodified PC laboratories on campus, or using the students' own devices.

When used as a tool for training, the VCC promotes development of skills that have a high level of transferability; Students can use the same environment in the safety of a virtual cluster without putting a production system at risk.

A. Implementation in the classroom

The OSCAR software that was originally used in the module has been replaced with the VCC. The students, working in groups, will build the cluster by installing CentOS 7, the Docker runtime and the VCC middleware on the physical machines. The VCC container will deploy a Torque/PBS cluster, including an MPI implementation and a C compiler. This environment is sufficient to complete the benchmarking task for assessment. Since the host OS is decoupled from the cluster environment, it is possible to use a newer version of the OS that avoids compatibility issues introduced by newer generations of hardware.

This strategy is able to mitigate many of the problems that were identified with the existing teaching method. Firstly, the reliance on dedicated hardware in order to deliver the module is reduced, as the VCC containers can be deployed on cross-platform devices. This should represent a net increase in reliability as university maintained hardware can be used, rather than requiring a dedicated lab to be built that may or may not have a formal programme of upgrade and replacement of equipment. It also reduces the amount of resource required per student, as a single machine can host many virtual nodes. This is acceptable in the training context, as the demonstration codes and codes that students create are not likely to significantly stress the system and will demonstrate scaling when increasing the number of processes beyond the number of physical CPUs (to the extent required in order to fulfil the learning outcomes). The advantage of being able to demonstrate scheduling interactions, queue management and use of the associated tools to interact with jobs on a seemingly large system outweighs the potential performance limitations from oversubscribing many virtual nodes on a single machine.

Secondly, the middleware has been designed so that it is by default pitched at the right skill level for learners, without having to dive too much into the low level networking and other implementation details. The installation process loosely mimics that of OSCAR in terms of high level steps. Some steps are removed and replaced with automatic mechanisms, typically those that are unnecessarily repetitive such as MAC address collection. The automation is not compulsory and is implemented in such a way as not to obfuscate the respective process - the knowledge on how to perform this process is not confined to a "black box". However, this automation offers many advantages: the student can deploy a working environment first, and dissassemble it later to learn how it functions, providing a compromise between the amount of time spent performing configuration before the system can be used for productive work. In addition, it reduces the amount of time that is needed to spend on the configuration stage of the project. This will help alleviate the *critical path* problem that has been present in the module.

Finally, the same resource providing a container runtime can be adapted to deliver training for a wide range of cluster based applications simply by changing the container that is executed by the student. This will allow a mixed mode course to be designed where the student is not subject to constant reconfiguration of the allocated resource.

VI. Evaluation

The implementation outlined in the previous section has been utilized for the 16/17 year of the *Parallel Computer Architectures* undergraduate module at the University of Huddersfield.

The purpose of this evaluation was to assess the usability of the VCC in comparison to the OSCAR tool, and gain an insight into the level of knowledge attained by the students during the course module. The first term of the undergraduate and postgraduate modules are both concerned with cluster technologies, and have significant overlap in terms of the lecture and laboratory activities. Therefore, it was decided to redesign the undergraduate module around VCC whilst retaining the original programme for postgraduate students, allowing for a comparison to be drawn as part of this study.

A. Usability

The International Organization for Standardization defines usability as ”The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.” [17]. The effectiveness, efficiency and satisfaction factors are affected by the users using the product, what they are trying to do with the product and the context in which the product is used. Usability is distinct from the capabilities and functionality of the product.

The System Usability Scale (SUS) is a quick and reliable tool to measure the usability of a system[18]. It provides a 10 question survey, each response consisting of a 5 point scale from strongly agree to strongly disagree. In order to analyze a SUS score, it can be converted to a percentile rank. The SUS has good reliability and generates reliable results based on small sample sizes. It does not diagnose the cause of any usability problems within a system, only indicating that one may exist. It has stood the test of time, with subsequent studies of this method suggesting that it still effectively distinguishes unusable and usable systems despite changes in technology[19][20]. It has also been recognized that the SUS, whilst originally intended to measure only one dimension, provides a measure of both usability and learnability components[21].

B. Skills Audit

In order to gain insight into the development of key skills during the module, a skills audit survey was devised. This consists of 12 skills that the student must rank their proficiency in that skill based on a scale of 0 to 5, with 0 being no knowledge at all and 5 being an expert in that skill. The skills cover a range of basic knowledge surrounding PC hardware, to advanced knowledge such as code parallelization strategies. It is expected that at the start of the module, most students will hold only basic knowledge. The same survey will be administered at the end of the module and the relative difference will give an informal understanding of the change in the student’s own perceived level of skill.

It has been suggested that the measure of usability can be affected by the experience of the user when administering the SUS[22]. This effect is characterized by users with more experience using a product tending to provide more favourable SUS scores over users who have no experience, regardless of the type of product being tested. Therefore, the skills audit can be used with the SUS study in order to observe this effect.

C. Method

At the start of both the undergraduate and postgraduate modules, the skills audit survey was administered to establish the perceived skill of the incoming students. At the end of the module, the skills audit was administered again in the exact same manner. The average of each response per question is used to compare between the first and second surveys.

The SUS survey was administered at the same time for both modules: at the end of the first term where the activities

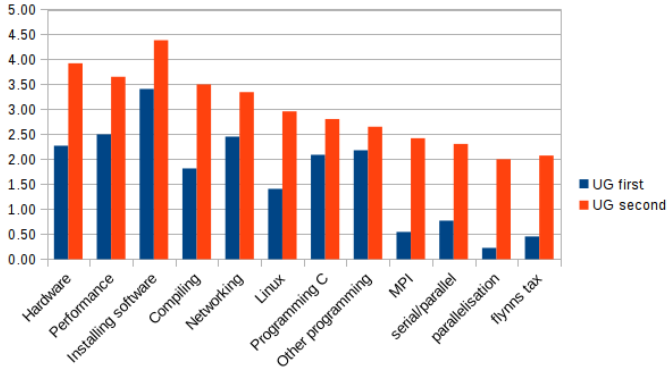

Fig. 2. Skills audit, undergraduate using VCC

are common for both undergraduate and postgraduate module, performed in accordance with the methodology set out in [18]. The raw SUS results are scores and must be converted into a percentile rank for a useful comparison. The SUS analysis will be conducted on the two factors identified in [21] to extract data on both usability and learnability parameters.

In addition, some ”fun” questions were added to the end of the paper in order to illicit subjective and free text comments from the students about their experience on the module and the tool that they used.

D. Results and Discussion

The results of the skills audit are presented in Figure 2 for undergraduate module using the VCC tool and Figure 3 for postgraduate module using OSCAR tool. It can be noted that both groups of students held their level of knowledge around the basic and intermediate skills (1-8) in higher regard than the advanced parallel and distributed specific skills (9-12). As expected, the postgraduate students on average both entered and exited with a perceived higher degree of proficiency in these skills.

However, when comparing the relative difference between the advanced skills, the undergraduate and postgraduate responses are much more aligned. This is expected as none of the students are expected to hold these skills, and will learn them during the module. The results suggest that the attainment and development in these skills, required in order to meet the learning outcomes of the module, is comparable whether the OSCAR or VCC tool is used. Therefore, the VCC tool is a suitable replacement for the OSCAR tool.

The SUS results are presented in Figure 4. At first glance, the VCC tool appears to have slightly more favourable responses on the SUS scale. The average SUS score of 68 is indicated in the figure. It can be seen that both tools are considered ”below average” compared to this line, which falls within the acceptable but marginal range as defined by [19]. Expressed as a percentile rank, this places the VCC score of 62.4 in the 30th percentile, whilst the OSCAR score of 57.5 is placed in the 20th percentile[23].

A comparison of the SUS scores shows that overall usability of the VCC tool is equal to and in some respects exceeds

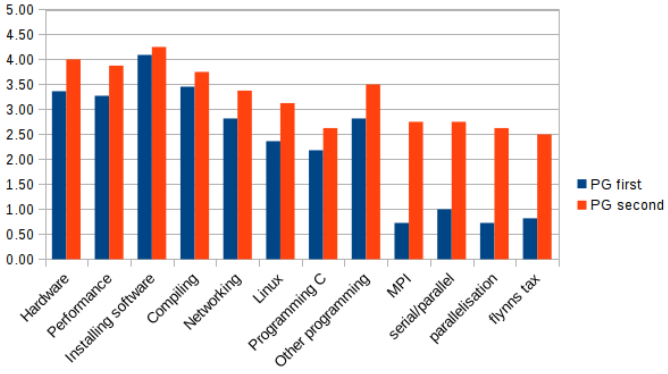

Fig. 3. Skills audit, postgraduate using OSCAR

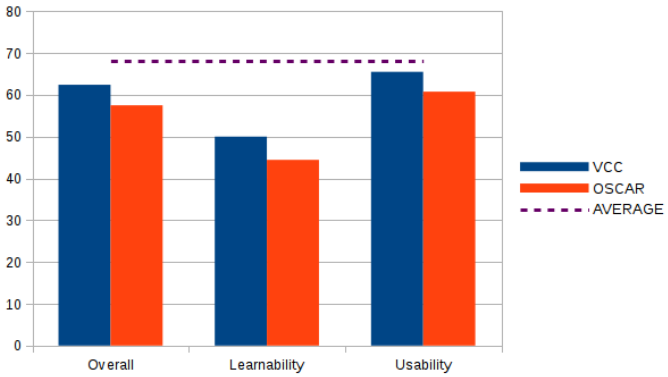

Fig. 4. Calculated SUS scores

that of the OSCAR tool. The usability of both these tools is below the average obtained from a large body of SUS scores evaluating a variety of systems. Decomposed into both factors, it can be seen that the learnability score is much lower. The questions relating to the learnability factor ask *"I think that I would need the support of a technical person to be able to use this system"* and *"I needed to learn a lot of things before I could get going with this system"*. The low score likely reflects the inherent difficulty of the subject matter and the fact that both tools are used to conduct complex processes and require a significant learning curve. However, the VCC tool consistently scores higher in both respects than the OSCAR tool, suggesting that the student experience was slightly better. A particularly interesting anecdote is that the VCC uses a Command Line Interface (CLI) whilst OSCAR provides a Graphical User Interface (GUI).

When considering both the SUS scores and the skills audit together, the experience of the students (inferred through the skills audit) on both modules is comparable, thus significant skewing of the SUS scores was not observed due to this effect. The undergraduate students were able to attain a proportional increase in proficiency using a tool that was perceived to be easier to use and easier to learn.

The free text responses agree with this conclusion, where students expressed frustration with the fact that a configuration error or misunderstanding usually resulted in a complete reinstallation of the cluster with OSCAR. Unexpected errors were experienced by all students, but those using the VCC commented that it was quick to resolve problems.

VII. Summary

The *Parallel Computer Architecture* modules at the University of Huddersfield are a popular choice among Computing and Engineering students. They have been delivered for several years with the aim to give a broad understanding and foundation in parallel and distributed computing concepts. The increasing student numbers is encouraging: demonstrating interest in these technologies and the perception that they will be useful skills for future industrial or academic deployment.

However, we identified hardware and software limitations in the teaching methods that have a negative impact on the long term sustainability of delivering these courses. This experience influenced the development of a new tool that would be able to meet the learning objectives and the technical constraints. This new toolkit is based on the Virtual Container Cluster (VCC), that allows any machine running Docker to virtualize large cluster environments.

The tool has been successfully deployed in the 16/17 academic year for undergraduate students and an analysis of the usability and learnability show that it is as good, and in some respects better than the previous tool OSCAR. Students achieved the same level of proficiency in key skills required to meet the learning outcomes using a tool that was perceived to be easier to use and learn.

This alleviates concerns surrounding the reliability of hardware in the laboratory, sustainability of the software and the ability to adapt the computing resources to deliver many different HPC technologies without significant reconfiguration effort. It also opens up opportunities to exploit other resources due to the cross-platform nature of Docker - supporting the deployment of the Linux cluster environment on Windows and Mac hosts. In this way, each student can take home the same capabilities for training that they can access in the laboratory.

The VCC is released as open source software and is available on GitHub[24].

VIII. Future work

A quantitative analysis of the processes involved in performing the tasks should be considered in order to formally demonstrate that the VCC functionality (rather than usability) is improved over previous tools and appropriate to meet the learning outcomes.

It is planned to develop and publish a range of exercises along with the VCC toolkit to create a cross-platform turn key learning environment for HPC.

References

[1] E-infrastructure Leadership Council, "Uk participation in high performance computing (hpc) at the european level," June 2012.

[2] Universities and Colleges Admissions Service, "Ucas directory, 2014 entry," June 2014.

[3] I. Habib, "Virtualization with kvm," *Linux Journal*, vol. 2008, no. 166, p. 8, 2008.

[4] D. Merkel, "Docker: Lightweight linux containers for consistent development and deployment," *Linux J.*, vol. 2014, Mar. 2014.

[5] "Openvz virtuozzo containers." https://openvz.org/Main_Page.

[6] "Linux containers." https://linuxcontainers.org/.

[7] "Docker." https://www.docker.com/.

[8] "Singularity." http://singularity.lbl.gov/.

[9] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and linux containers," *technology*, vol. 28, p. 32, 2014.

[10] J. Higgins, V. Holmes, and C. Venters, *High Performance Computing: 30th International Conference, ISC High Performance 2015, Frankfurt, Germany, July 12-16, 2015, Proceedings*, ch. Orchestrating Docker Containers in the HPC Environment, pp. 506–513. Springer International Publishing, 2015.

[11] V. Holmes and I. Kureshi, "Developing high performance computing resources for teaching cluster and grid computing courses," *Procedia Computer Science*, vol. 51, pp. 1714–1723, 2015.

[12] J. Mugler, T. Naughton, S. L. Scott, B. Barrett, A. Lumsdaine, J. M. Squyres, B. des Ligneris, F. Giraldeau, and C. Leangsuksun, "Oscar clusters," in *Linux Symposium*, p. 387, 2003.

[13] "Status - oscar project." http://svn.oscar.openclustergroup.org/trac/oscar/wiki/Status.

[14] P. M. Papadopoulos, M. J. Katz, and G. Bruno, "Npaci rocks: Tools and techniques for easily deploying manageable linux clusters," *Concurrency and Computation: Practice and Experience*, vol. 15, no. 7-8, pp. 707–725, 2003.

[15] "Welcome to apache hadoop." http://hadoop.apache.org/.

[16] J. Higgins, V. Holmes, and C. Venters, "Autonomous discovery and management in virtual container clusters," *The Computer Journal*, vol. 60, no. 2, pp. 240–252, 2016.

[17] I. O. for Standardization, *ISO 9241-11: Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs): Part 11: Guidance on Usability*. 1998.

[18] J. Brooke *et al.*, "Sus-a quick and dirty usability scale," *Usability evaluation in industry*, vol. 189, no. 194, pp. 4–7, 1996.

[19] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale," *Intl. Journal of Human–Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008.

[20] K. Orfanou, N. Tselios, and C. Katsanos, "Perceived usability evaluation of learning management systems: Empirical evaluation of the system usability scale," *The International Review of Research in Open and Distributed Learning*, vol. 16, no. 2, 2015.

[21] J. R. Lewis and J. Sauro, "The factor structure of the system usability scale," in *International conference on human centered design*, pp. 94–103, Springer, 2009.

[22] S. McLellan, A. Muddimer, and S. C. Peres, "The effect of experience on system usability scale ratings," *Journal of Usability Studies*, vol. 7, no. 2, pp. 56–67, 2012.

[23] J. Sauro and J. R. Lewis, "When designing usability questionnaires, does it hurt to be positive?," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2215–2224, ACM, 2011.

[24] "hpchud/vccjs: A framework for building hpc and parallel containers - github." https://github.com/hpchud/vccjs.