Middleware for Building Virtual Resources on Inter-Cloud

Shigetoshi Yokoyama*, Yoshinobu Masatani*, Tazro Ohta†, Osamu Ogasawara‡, Nobukazu Yoshioka*, Kento Aida*

*National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, JAPAN
†Database Center for Life Science, Yata 1111, Mishima, Shizuoka, 411-8540, JAPAN
‡National Institute of Genetics, Yata 1111, Mishima, Shizuoka, 411-8540, JAPAN

***Abstract*—With developments in information technology and big data analytics, demand for resources, servers, and data to be distributed over multiple cloud platforms is increasing. Inter-Cloud, which is a federated cloud computer environment, provides a promising approach to meet this demand. However, building the Inter-Cloud requires expert knowledge and specialized skills in network/server administration, and is technically challenging. In this paper, we propose an overlay cloud architecture for virtual cloud computing in Inter-Cloud. In the proposed architecture, the middleware, Virtual Cloud Provider (VCP), overlays virtualized computing resources on physical cloud computing resources using the Linux container technology, and overlays virtual network resources on a network, such as the Internet. The VCP enables the use of a customized computing environment, e.g. application software and required OS/libraries on allocated resources, and configures the virtual network between the resources. A case study which considers DNA sequencing software shows how the VCP can be deployed, and used to reproduce results.**

Keywords—Cloud Computing; Inter-Cloud; Container; Bioinformatics;

I. Introduction

Cloud computing has been widely used for enterprise applications, and there is a growing interest in the academic community in the use of cloud computing for scientific applications. The performance of cloud platforms in scientific applications has been investigated in numerous studies [1][2][3]. Public cloud platforms, e.g. Amazon EC2 [4] and Microsoft Azure [5], have also been used to run scientific applications. For example, the 1000 Genomes Project uses Amazon Web Service for storing and analyzing genome data [6].

Such scientific applications are usually run on a single cloud platform; however, the performance and reliability when using a single cloud platform is limited. Furthermore, in big science applications, data are produced and archived from many sources, e.g. experimental devices, sensors, databases, which are located around the world. It is not practical to collect and analyze all data on a single cloud platform. Inter-Cloud [7] presents a solution to the limitations of a single cloud platform. Inter-Cloud consists of multiple cloud platforms, distributed across geographically diverse locations and operated by independent providers, and it enables the coordination of resources and distribution of workloads among different cloud platforms. The user can then create a customized computing environment over multiple cloud platforms.

However, configuring a user-customized computing environment in Inter-Cloud is not as easy a task as in a single cloud platform. The user needs to control virtual machines (VMs), deploy application software and their dependencies (binaries and libraries) on VMs, and configure virtual networks among VMs over multiple cloud platforms. The above configuration processes require expert knowledge and advanced technical skills for network and computer administration. Software is available to support the above configurations, e.g. RightScale [8] and OpenNaaS [9]. However, the existing software only partially supports these configurations, and the user still needs to perform time-consuming work to configure the customized computing environment in Inter-Cloud.

Cloud computing is widely used in bioinformatics applications. For example, in DNA sequencing, a researcher runs a pipeline (or workflow), consisting of a chain of processing elements, to determine the order of nucleotides in a DNA molecule. It is common for researchers in the bioinformatics research community to make the databases of these DNA sequences publically available, e.g. DDBJ, NCBI and EMBL-EBI [10], when they publish research papers. Furthermore, publishing source code for the software used to determine the DNA sequences is also becoming common practice. Sharing both the results and the software among researchers significantly improves the reproducibility of the published results. However, the reproducibility of the results also depends on the underlying software environment, e.g. the operating system, libraries and binaries. In other words, we cannot guarantee the reproducibility between two computers with different underlying software environments, even when we build software from the same source code on both computers. This makes it hard to reproduce the results of computer experiments.

In this paper, we propose an overlay cloud architecture for building virtual clouds in Inter-Cloud. In the proposed architecture, the middleware, Virtual Cloud Provider (VCP), overlays virtual computing resources on physical cloud computing resources using the Linux container (or container) technology and overlays virtual network resources upon network such as the Internet. VCP automatically controls virtual machines (VMs) or bare-metal machines (BMs) through APIs on multiple cloud platforms, deploys application software or its dependencies (binaries and libraries) on VMs/BMs using the container technology, and configures virtual networks among VMs/BMs over multiple cloud platforms. The users can easily configure his/her customized computing environment, or the virtual cloud, in Inter-Cloud by defining the profile of the computing environment. The example of DNA sequencing shows how VCP enables the user to develop a customized computing environment that can produce reproducible results. To the best of our knowledge, this is the first paper that demonstrates a configuration of the overlay cloud middleware for a customized computing environment with reproducibility for use with DNA sequencing.

The remainder of this paper is organized as follows. Section II presents the background for this paper. Section III presents the proposed overlay cloud architecture and the implementation of VCP, and Section IV considers a case study of DNA sequencing. Finally, Section V summarizes the contributions of this paper and outlines future work.

II. Background and Related Work

In this section, we introduce Inter-Cloud and review the overlay architecture for use with Inter-Cloud.

A. *Inter-Cloud*

Inter-Cloud is an architecture for organizing a cloud of clouds. It utilizes multiple cloud platforms to offer enhanced capabilities. Examples of Inter-Cloud being used for business applications have been discussed [7]. These include guaranteed end-to-end quality of service, enhanced convenience through service cooperation and service continuity. For scientific applications, Inter-Cloud can be used as a distributed computing platform to utilize computing, storage and network resources over the Internet.

Progress has been made in standardizing platforms (or providers) to allow them to interoperate with each other [11][12][13]. The standards usually concern models and protocols among cloud providers to enhance disaster recovery capabilities, service continuity and for cloudburst use. Testbeds for investigating inter-operation have also been discussed [14]. Although such an approach offers numerous benefits, in reality very few implementations conform to these standards. This is because the implementation would have a huge initial cost and would require frequent upgrades of middleware on cloud platforms.

B. *Inter-Cloud by Overlay Architecture*

The other approach is to use the overlay network for federating heterogeneous cloud platforms. Recent virtualization technologies make this approach feasible. For example, container technology, such as Docker [15], enables the deployment of applications in containers which run on heterogeneous cloud platforms. Docker is a container management system, and most public cloud providers currently support it. It helps applications move between multiple clouds and realize inter-cloud environments.

The workflow execution management system, Skyport [16], enables container-based workflows to run on multiple cloud platforms. The development of Skyport was based on the workflow management system, AWE/Shock. When a job is assigned to a computing resource (AWE worker), the AWE worker automatically downloads the input files and Docker container images needed to run the job. Skyport automates application deployment to multiple cloud platforms; however, it does not have a mechanism for configuring the virtual network among computing resources. Furthermore, Skyport assumes that the AWE/Shock framework is already installed on the cloud platforms. Thus, the user needs to manually install AWE/Shock if it is not available. It also imposes the limitation that only software which is compatible with Skyport can be used.

In the proposed overlay cloud architecture, VCP automatically configures not only the computing resources but also the virtual networks between them. Furthermore, VCP eliminates the dependency on preinstalled software, such as the workflow management system, by using the nested container mechanism described in Section III.A.

III. Proposed Architecture

In this section, we present the proposed overlay cloud architecture. The middleware, VCP, is the core technology for implementing the proposed architecture. We also present the design and implementation of the VCP.

A. *Overlay Cloud Architecture*

Fig. 1 presents an overview of the overlay cloud architecture. As can be seen, the overlay cloud is built on computing resources, e.g. virtual machines (VMs) or bare-metal machines (BMs), and network resources provided by real cloud providers, such as private/public cloud providers. Here, the user is able to access the resources on demand. We assume that the resources are connected with an academic network, such as SINET, and VPN services [18] are available on this network.

The primary purposes of the VCP are the deployment of software, including applications and system software, and the creation of a virtual network on VMs/BMs over multiple cloud platforms. To this end, VCP uses the container technology, Docker, to deploy software and connects the containers via overlay networks over virtual networks provided by the real cloud providers. We can assume that the overlay cloud architecture works on the current cloud platforms in the real world. Most public cloud providers currently support Docker engines and the user is able to run the same container images on different resources in the different cloud providers. In addition, we assume that the overlay networks can be created using Linux standard network services like Open vSwitch. Under these assumptions, the user is able to deploy containers on Inter-Cloud over real cloud providers without any prior negotiations with the providers.

The software deployment process consists of two phases: the base and application phases. We assume that the user runs a workflow of an application on distributed computing resources. It is common that the workflow application is executed using

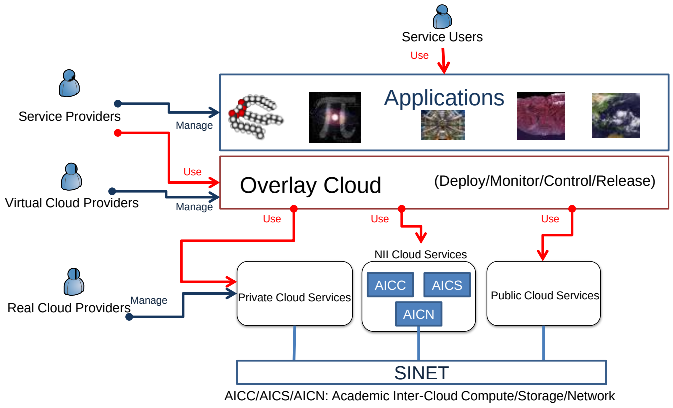

Fig. 1. Overlay cloud architecture

cluster management software, e.g. a job scheduler, in the distributed computing systems. In the base phase, containers are deployed for cluster management software. We refer to these containers as base containers. In the application phase containers are deployed for applications, called application containers. Existing cluster management software, e.g. Mesos [17], allows containers to be run. The application containers are executed with the cluster management software in the base containers. The overlay cloud architecture utilizes nested containers (a container within a container). Performance issues arise depending on the configuration of the nested virtualization layers. Our preliminary experiments show that the performance degradation due to the use of nested containers is negligible.

The deployment of virtual networks in Inter-Cloud depends on the underlying network architecture. In the typical scenario where the user runs applications on virtual machines in public clouds, VCP creates a virtual network between resources using conventional technology, e.g. IPsec. The communication performance may be highly degraded in this scenario, if secure communications are used. The other scenario is that the user runs applications on bare-metal machines (or bare-metal clouds) that are directly connected to the academic network with a high-speed VPN service, e.g. SINET L2VPLS [18]. We can achieve high-performance and secure communication between computer resources by using a high-speed VPN service.

B. Virtual Cloud Provider

VCP is the core technology for implementing an overlay cloud architecture. Here, we describe how the VCP meets the requirements of applications running in the overlay cloud architecture. We assume that the following requirements should be fulfilled:

- Req-0) The user is able to build a customized computing environment, or a virtual cloud, for the user application.
- Req-1) The user is able to use resources on major cloud platforms in the virtual cloud on demand.
- Req-2) No prior negotiations are needed among cloud providers to build the virtual cloud.
- Req-3) The user is able to use the full capabilities of the physical resources for running the user application.
- Req-4) The virtual cloud should be scalable to the size of the problem.
- Req-5) The virtual cloud should support applications from various scientific fields.
- Req-6) The user is able to migrate applications to computing resources or sites, where huge databases or supercomputers are operated, in order to minimize data transfer time on the wide area networks.
- Req-7) The user is able to run applications on confidential data in a secure manner.

We designed the VCP with the following criteria to fulfill the above requirements:

1) We improves the portability of applications by creating containers for applications and deploying these containers to resources in the virtual cloud. This fulfills Req-0, Req-1, Req-2, Req-3 and 5.
2) We improves the scalability of the virtual cloud by creating containers for cluster management software (base containers) and deploying the base containers to distributed resources in the virtual cloud. This fulfills Req-0, Req-4 and Req-6.
3) We connected base containers with a high-speed, secure VPN service in an academic network, e.g. SINET VPLS. This fulfills Req-3 and Req-7.

According to the criteria, the proposed VCP architecture consists of application container management, base container management, network management and container image management. In order to keep virtual cloud portability among various real clouds, VCP has to handle those cloud managements APIs to get resource for running base containers on which application containers can be run. In this architecture, application containers run in base containers. This container in container configuration is needed because of applications and virtual clouds portability. The processing performance impact due to the nested container set up can be said almost negligible from our evaluations.

Speaking of network management, it has two layers. One of them is application container networking. The other is base container level networking. It is better to have L2 connections among base containers because of make application container networks configuration remain flexible as much as possible.

The connection among base containers is realized by the connection among host computers which are allocated from various cloud providers. From network performance points of view, it is better to use VPN service from wide area network provider than using L2 over L3 tunnels among the hosts.

The network performance of the tunnel with IPsec for security is no practical for some applications from our evaluation. From this regard, the proposed network architecture is L2VPN for base container network layer and L2 over L3 tunnel for application container layer. The application containers network can be implemented on this L2VPN network among base containers.

It can be VXLAN tunnels because it is becoming popular in container network implementation in the communities like docker eco system.

C. *Implementation of the VCP*

We developed a prototype VCP, which provides four services:

1) *Cloud Resource Management*

The cloud resource management service allocates computing resources (VMs or BMs) in real cloud providers to base containers. The user is able to modify the resource configuration (adding/deleting resources) on demand.

2) *Base Container Management*

The base container management service determines suitable computing resources for the deployment of base containers and configures parameters when launching containers. It also configures network settings, upgrades base containers or deletes containers in the same cluster at once.

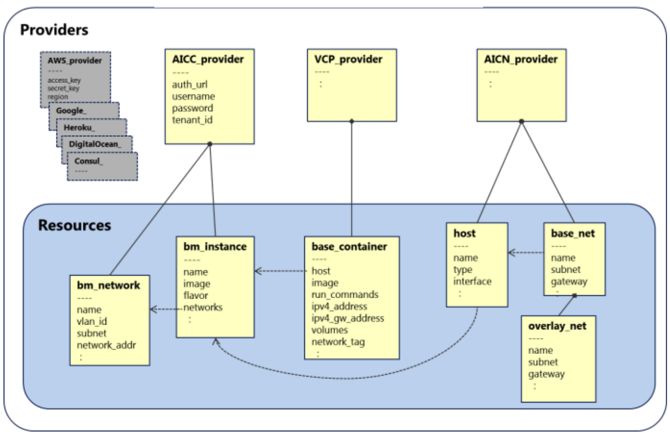

Fig. 2. Services available in Terraform

3) *Network Management*

The network management service configures virtual networks, or VLANs, that connect base containers. It also configures the gateway of VLANs.

4) *Base Container Image Management*

The base container image management service consists of two repositories. The container image repository archives images of base/application containers, and the template repository archives templates of application configurations.

The VCP uses an orchestration mechanism, which automates deployment, configuration and management of the virtual cloud for the user applications. We use open source software, Terraform, for this orchestration in the VCP. Terraform has the following advantages.

- Terraform has a plug-in mechanism that enables the user to add third-party services.
- Terraform enables the user to configure multiple services in a single configuration file.
- Terraform enables the user to manage not only IaaS resources but also PaaS/SaaS resources.

We implemented the following three services using the plug-in mechanism of Terraform. Fig. 2 shows the relationships between the services.

- AICC (Academic Inter Cloud Computing) Provider

The AICC provider is a cloud resource management service customized for our cloud platform, AIC. The AIC is a bare-metal cloud platform that offers IaaS resources for researchers in the National Institute of Informatics. The AICC provider enables the user to control computing resources in the AIC. The AIC is operated using OpenStack. The APIs for controlling computing resources in the AICC provider are compatible with those of OpenStack.

- VCP Provider

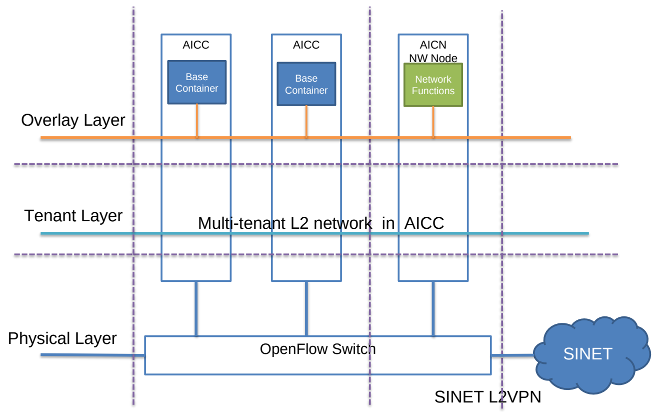

Fig. 3. Network layers in virtual cloud

The VCP provider is an implementation of the base container management service. It enables the user to control base containers in the Terraform framework.

- AICN (Academic Inter Cloud Network) Provider

The AICN provider is an implementation of the network management service. A network in a VCP's virtual cloud consists of multiple layers, as illustrated in Fig. 3. The overlay layer connects base containers, and the tenant layer connects VMs/BMs in a cloud platform. The AICN provider controls both overlay and tenant layers.

As shown in Fig. 2, the AICC provider calls AICC APIs in order to prepare BMs for base containers. The AICN provider configures base networks, which connect the BMs using AICN APIs. It also creates overlay networks in the base networks to connect base containers, that is, it makes clusters of base containers. Then, the VCP provider deploys base containers on the clusters.

IV. CASE STUDY OF BIOINFORMATICS APPLICATION

We conducted a preliminary experiment to build a virtual cloud for DNA sequencing applications using the VCP. Figure 4 shows the customized computing environment for DNA sequencing in the experiments. The researcher runs a workflow of multiple software tools to determine the order of nucleotides in a DNA molecule. Galaxy [19] is a widely used web-based platform for managing workflows for DNA sequencing (Fig. 4). It enables the user to import software tools from the workflow repositories, create a workflow and submit the workflow to computing resources. The computing resources are operated with the cluster management software as outlined in Fig. 4. We use Mesos/Aurora cluster management software in this experiment. Thus, jobs in the workflow submitted from Galaxy are executed through the Mesos/Aurora framework.

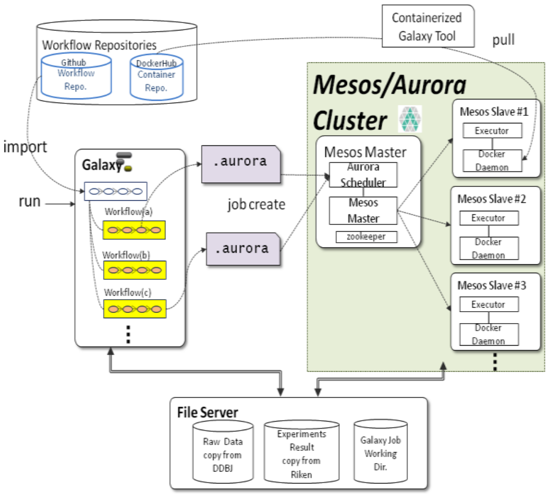

Fig.4. Computing environment for DNA sequencing application

In software deployment in the VCP, we created base containers for the Mesos/Aurora framework and Galaxy. The VCP deploys the base containers to computing resources in the cloud platforms, AIC. The file server in Fig. 4 allows application containers to access data in the server. Applications could be deployed to distributed computing resources in the Internet; thus, the file servers in the figure would be able to access distributed computing resources. There exist several options for the configuration of the file server, e.g. distributed file system, object storage or a network pipe. We used NFS for the file servers in the experiments. However, we leave the experiment with other implementations for future work.

Fig. 5 shows a workflow of the DNA sequencing application, RNA-Seq [20], which is used for analyzing the sequence of RNA in a cell. In the experiment, we created application containers for jobs in the workflow, “TopHat2”, “Cufflinks”, “Cuffmerge” and “Cuffdiff”. Then, we deployed base containers, the Mesos/Aurora framework and Galaxy, to four computing nodes (Intel Xeon E5-2670 2.60 GHz 8-core x2CPU, 96 GB memory) in AIC. Here, the Mesos master, the Aurora scheduler, Galaxy and file servers are deployed to the single node. The Mesos slaves are deployed to the other three nodes. The application containers are deployed in the Mesos/Aurora framework and the jobs are scheduled and executed on the Mesos slaves.

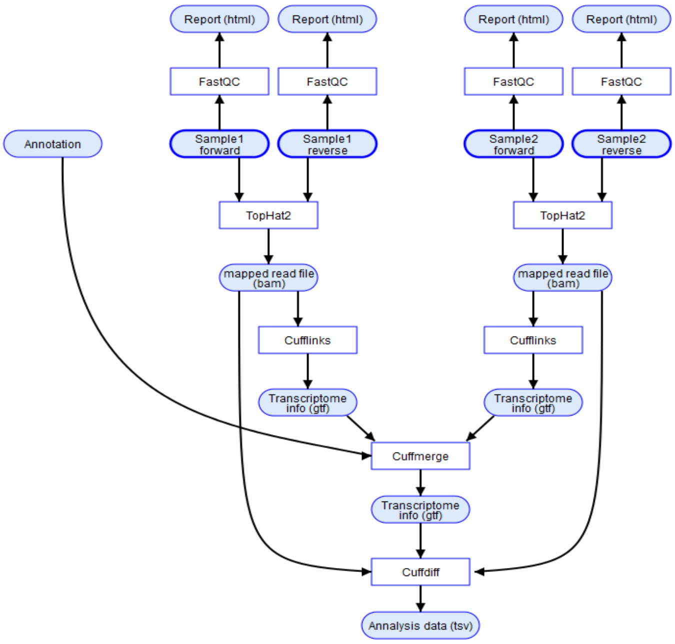

Fig.5. Workflow of RNA-Seq application

We confirmed that the VCP repeatedly deploys the above computing environment. The results indicate that the VCP enables the user to deploy the customized computing environment and that the results are reproducible. Figure 6 shows the execution time for RNA-Seq on the above computing environment. The size of the input file, “FastQC” in the workflow, is 12 GB. We parallelized the longest job, “TopHat2”, in the workflow, which resulted in an increase in the processing speed. However, the improvement in the performance is saturated at 24 parallels due to the performance bottleneck in the NFS servers.

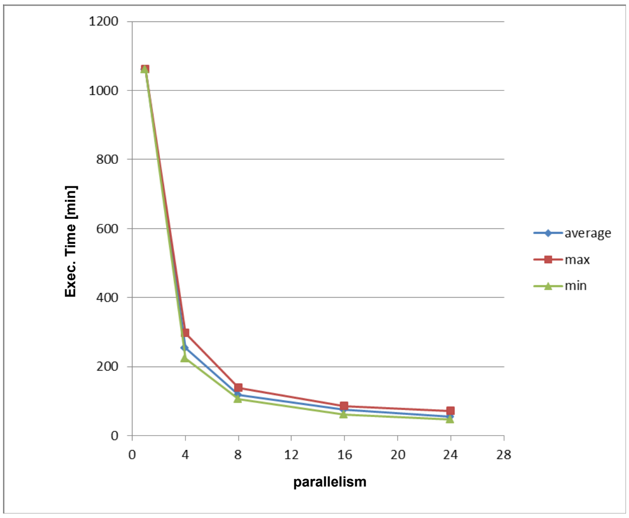

Fig.6. Execution time of the RNA-Seq application

V. Conclusions

We proposed an overlay cloud architecture for networking virtual clouds in Inter-Cloud. We developed the middleware, VCP, and presented a case study involving a DNA sequencing scenario. The preliminary experimental results show that the VCP enables the user to deploy the computing environment that produces reproducible results.

The case study presented in this paper is obtained from preliminary experiments which were run on a single cloud platform. We need to conduct more experiments with different settings. For example, by repeating the experiments with different cluster management software, e.g. Kubernetes [21]. Experiments on multiple clouds are also an area for future work.

In the preliminary experiments, the application framework schedulers determine how resources are used by the application containers, and application providers determine the base container allocation. In the inter-cloud situation, the allocation of resources has to be done dynamically for ease of use, resource efficiency and application performance. The design and implementation of the mechanism is our future work. There are further areas for consideration, such as how to select the optimal virtual network deployment process, which we leave for future work.

Acknowledgment

We thank Mr. Jun Nasuno for his support in the implementation of the VCP and the experiments. This work was supported in part by CREST, JST.

References

[1] G. Juve, E. Deelman, K. Vahi, G. Mehta, B. Berriman, B.P. Berman, and P. Maechling, “Scientific workflow applications on Amazon EC2,” Proceedings of the 2009 5th IEEE International Conference on E-Science Workshops, pp.59–66, 2009.

[2] K. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H.J. Wasserman, and N. Wright, “Performance analysis of high performance computing applications on the Amazon Web Services cloud,” Proceedings of the 2010 IEEE 2nd International Conference on Cloud Computing Technology and Science (CloudCom 2012), pp.159–168, 2010.

[3] A. Marathe, R. Harris, D.K. Lowenthal, B.R. de Supinski, B. Rountree, M. Schulz, and X. Yuan, “A comparative study of high-performance computing on the cloud,” Proceedings of the 22nd international symposium on High-performance parallel and distributed computing (HPDC ’13), pp.239–250, 2013.

[4] Amazon Web Service , http://aws.amazon.com/ (accessed on 12/18/2015)

[5] Microsoft Azure, https://azure.microsoft.com/ (accessed on 12/18/2015)

[6] 1000 Genomes, A Deep Catalog of Human Genetic Variation, http://www.1000genomes.org/

[7] Global Inter-Cloud Technology Forum, “Use Cases and Functional Requirements for Inter-Cloud Computing,” Technical Report, Global Inter-Cloud Technology Forum, 2010.

[8] RightScale, http://www.rightscale.com/ (accessed on 12/18/2015)

[9] J.I. Aznar, M. Jara, A. Rosello, D. Wilson, S. Figuerola, “OpenNaaS based management solution for inter-data centers connectivity”, IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom 2013), vol. 2, pp.45-50, 2013.

[10] International Nucleotide Sequence Database Collaboration, http://www.insdc.org/

[11] Open Cloud Computing Interface (OCCI), http://occi-wg.org/ (accessed on 12/18/2015)

[12] Open Grid Forum, Network Service Interface (NSI), https://redmine.ogf.org/projects/nsi-wg/

[13] Distributed Management Task Force, Inc. (DMTF), Open Virtualization Format (OVF), http://www.dmtf.org/standards/ovf/ (accessed on 12/18/2015)

[14] D. Bernstein, Y. Demchenko, "The IEEE Intercloud testbed - Creating the global cloud of clouds," Proceedings of the 2013 IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom 2013), vol.2, pp.45-50, 2013.

[15] Docker, https://www.docker.com/ (accessed on 12/18/2015)

[16] W. Gerlach, W. Tang, K. Keegan, T. Harrison, A. Wilke, J. Bischof, M. D'Souza, S. Devoid, D. Murphy-Olson, N. Desai, F. Meyer, "Skyport - container-based execution environment management for multi-cloud scientific workflows," Proceedings of the 5th International Workshop on Data-Intensive Computing in the Clouds (DataCloud), pp.25-32, 2014.

[17] The Apache Software Foundation, Apache Mesos, http://mesos.apache.org/ (accessed on 12/18/2015)

[18] S. Urushidani, S. Abe, K. Yamanaka, K. Aida, S. Yokoyama, H. Yamada, M. Nakamura, K. Fukuda, M. Koibuchi, and S. Yamada , “New directions for a Japanese academic backbone network”, IEICE TRANSACTIONS on Information and Systems, vol. 98, no. 3, pp.546-556, 2015.

[19] Galaxy, Data intensive biology for everyone, https://galaxyproject.org/ (accessed on 12/18/2015)

[20] Pitagora-Galaxy, Workflow RNA-seq 01, http://wiki.pitagora-galaxy.org/wiki/index.php/Workflow_RNA-seq_01 (accessed on 12/18/2015)

[21] Google, Kubernetes, http://kubernetes.io/ (accessed on 12/18/2015)