Enabling the Deployment of Virtual Clusters on the VCOC Experiment of the BonFIRE Federated Cloud

Raul Valin, Luis M. Carril, J. Carlos Mouriño, Carmen Cotelo, Andrés Gómez, and Carlos Fernández
Supercomputing Centre of Galicia (CESGA)
Santiago de Compostela, Spain
Email: rvalin,lmcarril,jmourino,carmen,agomez,carlosf@cesga.es

***Abstract*—The BonFIRE project has developed a federated cloud that supports experimentation and testing of innovative scenarios from the Internet of Services research community. Virtual Clusters on federated Cloud sites (VCOC) is one of the supported experiments of the BonFIRE Project whose main objective is to evaluate the feasibility of using multiple Cloud environments to deploy services which need the allocation of a large pool of CPUs or virtual machines to a single user (as High Throughput Computing or High Performance Computing). In this work, we describe the experiment agent, a tool developed on the VCOC experiment to facilitate the automatic deployment and monitoring of virtual clusters on the BonFIRE federated cloud. This tool was employed in the presented work to analyse the deployment time of all possible combinations between the available storage images and instance types on two sites that belong to the BonFIRE federated cloud. The obtained results have allowed us to study the impact of allocating different requests on the deployment time of a virtual machine, showing that the deployment time of VM instances depends on their characteristics and the physical infrastructure of each site.**

Keywords-Cloud computing; Federated clouds; Virtualization; Cloud platforms; Virtual clusters; IaaS; SaaS.

I. INTRODUCTION

The BonFIRE Project [1] supports experimentation and testing of innovative scenarios from the Internet of Services research community, specifically focused on the convergence of services and networks. BonFIRE operates a Cloud facility based on an Infrastructure as a Service delivery model with guidelines, policies and best practices for experimentation. A federated multi-platform approach is adopted, providing interconnection and interoperation between novel service and networking testbeds. BonFIRE currently comprises of 6 geographically distributed testbeds across Europe, which offer heterogeneous Cloud resources, including compute, storage and networking. Each testbed can be accessed seamlessly with a single experiment descriptor, using the BonFIRE API that is based on the Open Cloud Computing Interface (OCCI). Figure 1 shows details about resource offering on the different testbeds, which include on-demand resources.

The BonFIRE project is also studying the possible federation of the BonFIRE testbeds with a variety of external cloud facilities, such as those provided by FEDERICA or OpenCirrus. BonFIRE offers an experimenter control of available resources. It supports dynamically creating, updating, reading and deleting resources throughout the lifetime of an experiment. Compute resources can be configured with application-specific contextualisation information that can provide important configuration information to the virtual machine (VM); this information is available to software applications after the machine is started. BonFIRE also supports elasticity within an experiment, i.e., dynamically create, update and destroy resources from a running node of the experiment, including cross-testbed elasticity.

INRIA currently offers on-request compute resources in BonFIRE, allowing experimenters to reserve large quantities of physical hardware (162 nodes/1800 cores available). This gives experimenters flexibility to perform large-scale experimentation, as well as providing greater control of the experiment variables as exclusive access to the physical hosts is possible. Further control of network performance between testbeds is anticipated through future interconnection with Federica and GÉANT AutoBAHN. BonFIRE gives you control of your experiment, which is treated as a concrete entity in BonFIRE to manage your resources.

Some additional features implemented on the BonFIRE project are:

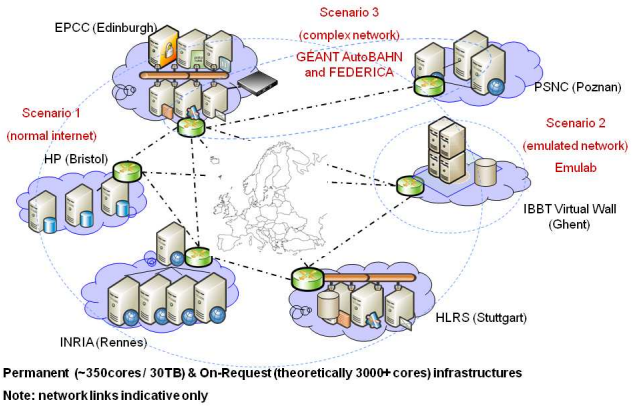

Figure 1. Representation of the BonFIRE infrastructure. Image obtained from [1].

 ISBN: 978-1-61208-216-5

- Saving compute disk images with your personal software stack, as well as storage resources.
- Sharing saved compute and storage resources.
- Sharing access to experiments with colleagues.
- Repeating experiments and sharing experiment descriptions for others to set up.
- Aggregated monitoring metrics at both resource level (e.g., CPU usage, packet delay, etc.) and application level for your VMs.
- Aggregated monitoring metrics at infrastructure level at selected testbeds.

Virtual Clusters on federated sites (VCOC) is one of the supported experiments of the BonFIRE Project [2]. Its main objective is to evaluate the feasibility of using multiple Cloud environments to deploy services that need the allocation of a large pool of CPUs or virtual machines to a single user (as High Throughput Computing or High Performance Computing). This experiment considers the deployment of virtual clusters with the propose of executing a radiotherapy application, developed in the eIMRT project [3], which calculates the dose for radiotherapy treatments based on Monte Carlo methods.

The VCOC experiment tried to answer different questions related to the usage of virtual clusters in distributed Cloud environments, analysing the advantages of deploying virtual clusters in a federated cloud. As part of the VCOC experiment a set of experiments have been proposed related to the time that the deployment and enlargement of a virtual cluster need to be operational as well as the influence that other simultaneous operations have on the process. The final objective is to get a better understanding about how to manage these virtual clusters to guarantee a reasonable time to solution or latency. A set of application probes have been chosen and they will help us to study the elasticity. The information provided by these probes will be used to monitor the performance of the application and to trigger the change in the size of the cluster. This information will be also combined with the information provided by the monitoring tools of BonFIRE.

The results and data acquired during the VCOC experiment should permit to develop policies and business rules to include in the applications under development at the institution which use the Software as a Service model.

In this paper, we introduce a description of the experiment agent developed by the VCOC experiment to manage the deployment of virtual clusters to the BonFIRE federated cloud as well as the required time to deploy individual instances with different configurations and images. The structure of the paper is as follows: In section II we describe the experiment agent and its main functionalities. A description of the error and log manager implementation is also shown in this section. The required time to deploy individual instances of the available virtual machine images in BonFIRE infrastructure is shown in Section III. Finally, the main conclusions of the paper are drawn in Section IV.

II. Description of the experiment agent

The BonFIRE project provides several ways to submit an experiment depending on the user preferences.

- The BonFIRE Portal (GUI) in a step-by-step manner [4].
- A command line client tool, such as Restfully, to interact with BonFIRE [5].
- A script for your experiment deployment, which can be automatically executed by, e.g., Restfully.
- The BonFIRE experiment descriptor, currently based on JavaScript Object Notation (JSON) [6].
- Raw HTTP commands via cURL [7].

In the VCOC experiment, we have developed an experiment agent to deploy, control and monitor the deployed experiments through a single interface. This agent was developed in Python and communicates with the experiment manager API using the httplib2 [8] library. Experiments are described in a JSON file specifying the necessary resources of the virtual clusters. After the submission of the experiment, the XML response is processed to obtain the basic information of the deployed resources. This information enables us to monitor the status of each virtual compute node and it is saved into a local file for future analysis. The main functionalities of the experiment manager are as follow.

- Experiment controller, controls the repetition of the experiments and communicates with the Experiment Manager using the API.
- Experiment status, measures accurately the time for each step in the work–flow and store this information.
- Experiment failures, detects the fail of an experiment and its resubmission.
- Experiment descriptors, accepts an experiment description in JSON and the number of repetitions as input.
- Multi-experiment, supports the sequential execution of several experiment descriptions.
- Simultaneous experiment, controls several experiment descriptions simultaneously.
- Random Experiment, generates random deployment requests which follow a defined time pattern to introduce load on the infrastructure.
- Sequential experiments, executes and controls two experiment descriptions with a predefined delay between them.
- Experiment accounting, records the BonFIRE accounting units for each experiment description.

A work–flow diagram of the experiment agent is depicted in Figure 2. This figure shows three main levels after a experiment submission. First, we evaluate the status of the experiment until the experiment has been deployed; this means the JSON experiment description has been processed

 ISBN: 978-1-61208-216-5

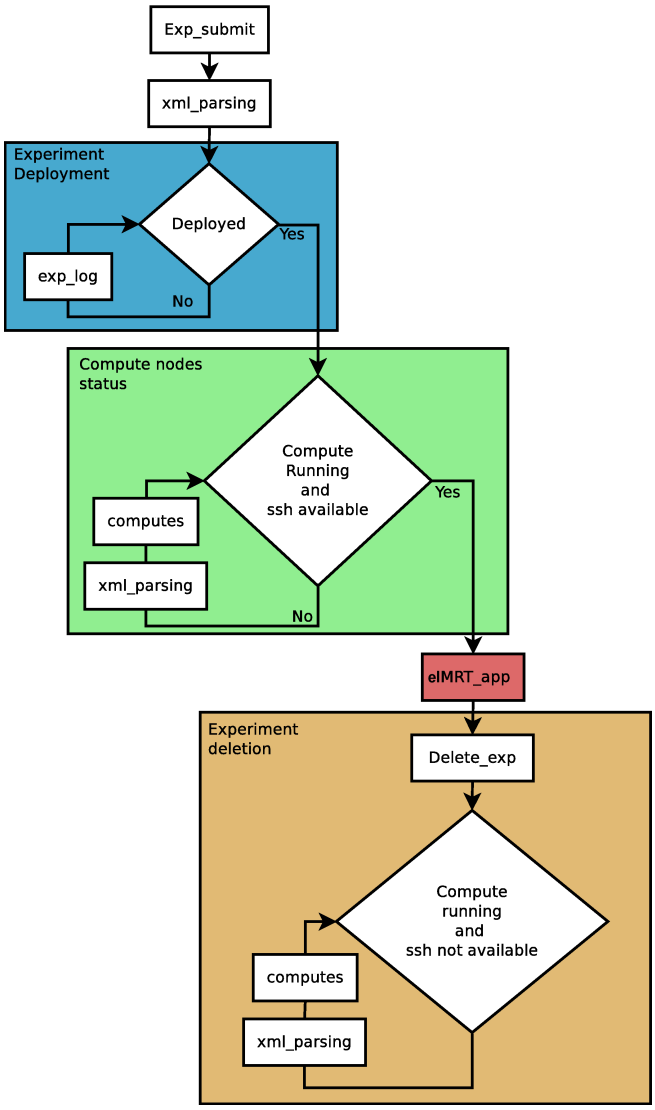

Figure 2. Workflow diagram of the experiment agent.

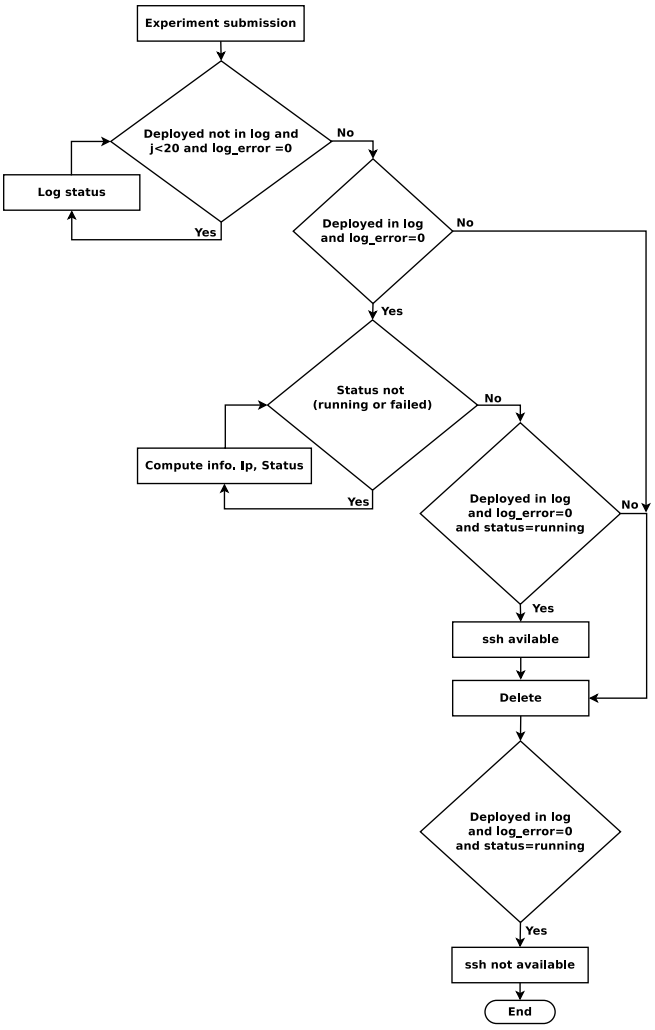

Figure 3. Workflow diagram of the error management implemented in the experiment agent.

by the Broker interface. After that, we have to wait for the allocation of the requested resources in the Broker layer of the infrastructure. This is the second level of the experiment agent, which evaluates if the required resources are running and ssh–available. When the requested resources are ssh–available, the eIMRT application will be executed. The last level of the experiment submission is the experiment deletion, which is carried out when the eIMRT application has finished. The experiment agent considers the deletion completed when the experiment is not ssh–available and the deleted resources reached the status DONE.

Two important functionalities implemented on the experiment agent are the error management and the measurement of the required time to complete each level described above. The error management has been developed to perform unattended deployment of experiments, taking into account possible errors or delays during the deployment of the experiment. Figure 3 shows the flow diagram of the error management that has been implemented in the experiment agent. First, after the experiment submission we have to evaluate if the experiment has been correctly deployed or any error has occurred. The experiment is deployed when the log status information provided by BonFIRE, after checking the experiment definition, returns the status deployed. If some error exists, the experiment will be resubmitted again. Otherwise, if there are not errors the experiment agent will evaluate the status of the requested VMs until they are ssh–available. When the VMs are available, the eIMRT application is executed and the experiment is destroyed when the eIMRT application is completed. are not ssh–available.

The measured times implemented on the experiment agent script try to provide information about the necessary time to deploy/destroy the computational resources available from

 ISBN: 978-1-61208-216-5

the submission/destruction of the experiments. Therefore, the experiment agent saves a timestamp value when the experiment is submitted and after its submission, a new timestamp value is saved when the computational resources are ssh–available. The difference between these two timestamp values provides the necessary time to deploy the resources requested on the JSON file. From the obtained times a first idea of the quality of service can be sketched if we want to use the infrastructure as a service. Finally, the experiment agent also measures the necessary time to destroy the experiment from the destruction request until the VMs are not ssh-available and the deleted resources reached the status DONE.

The last functionality implemented on the experiment agent, that we think must be highlighted in this description, is the possibility of deploying random experiments which follow a predefined pattern obtained from the accounting system of the Finisterrae supercomputer hosted in CESGA [9]. This functionality has been developed to introduce random noise into the BonFIRE infrastructure when we have exclusive access and therefore, we need to evaluate the impact of scheduling new experiments simultaneously.

This functionality returns a histogram with a discrete experiment submission probability such as is depicted in Figure 4, where the experiment submission probability for each 30 minutes of a day of the week is represented. Therefore, if we want to deploy random experiments during one day each 30 minutes at the same time that we are deploying our experiments, we will need to indicate the desired number of random experiments that they will be deployed during one day and the experiment agent will distribute the desired number of experiments taking into account the probability distribution. Figure 5 shows an example of the distribution of 100 random experiments taking into account the histogram depicted in Figure 4.

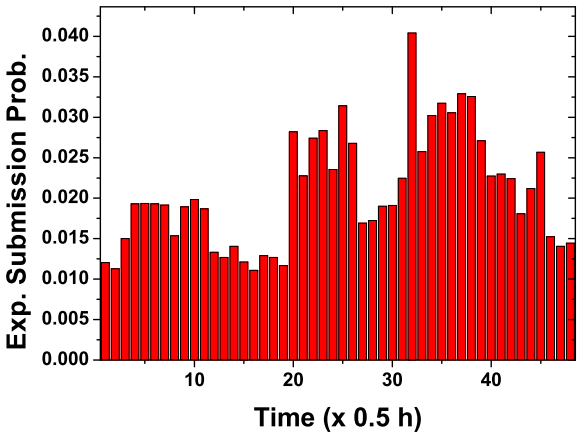

Figure 4. Experiment submission probability each 30 minutes of a day of the week.

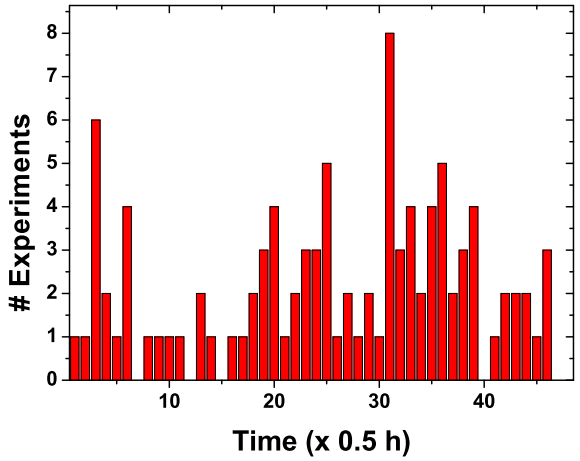

Figure 5. Example of the distribution of 100 random experiments taking into account the histogram depicted in Figure 4.

III. Results of the deployment time on EPCC and INRIA BonFIRE sites

The BonFIRE federated cloud has predefined resources, storage images, instances and networks, which are available for users. Table I and Table II show the resources, virtual machine images and instance types, available on EPCC and INRIA BonFIRE sites. Furthermore, two network resources are also available in these two sites enabling us choosing between either an Internet connection or a WAN connection. In this work, we have studied the deployment time of all possible combinations between the available storage images and instance types on EPCC and INRIA BonFIRE sites. The main goal of this study is to analyse the impact on the deployment time of allocating different requests.

The methodology adopted to carry out the experiment was based on the deployment on each site of each combination of storage-instance type. Each one of these combinations was deployed 10 times using the experiment agent. The final value to calculate the deployment time is equal to the mean of the 10 values provided by the experiment agent from the experiment submission until the VM is ssh-available, such as it was described in Section II.

Figure 6 depicts the deployment time for each storage-instance type combination on EPCC BonFIRE site. The obtained results show that DebSqV3 and DebSq2GV3 images have similar deployment times between 50-100 seconds, independently of the instance type. The deployment time increases in a rate similar to the size of disk image of the VM, represented on the figure by a red line. Therefore, the 10 GB storage image has the largest deployment time higher than 300 seconds independently of the instance type.

The deployment time for each storage-instance type com-

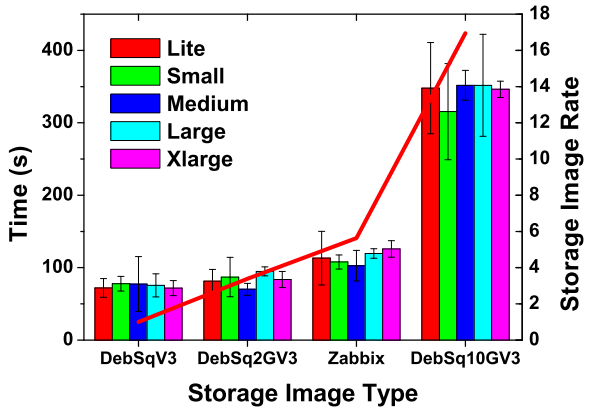

Figure 6. Deployment time for each storage-instance type combination on EPCC BonFIRE site. The size rate of the storage images with respect to the DebSqV3 image (red line) is also depicted for comparison reasons.

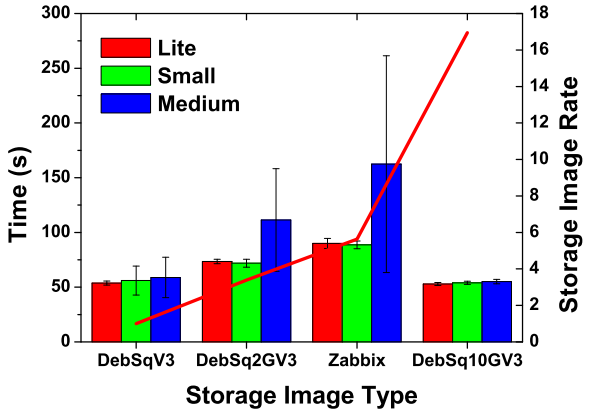

Figure 7. Deployment time for each storage-instance type combination on INRIA BonFIRE site. The size rate of the storage images with respect to the DebSqV3 image (red line) is also depicted for comparison reasons.

bination on INRIA BonFIRE site is depicted in Figure 7. The deployment time of DebSqV3 storage images on INRIA is around 50 seconds, independently of the instance type. For DebSq2GV3 and Zabbix storage images, the deployment time depends on the instance type with larger values for Medium instances around 100 seconds for DebSq2GV3 and 150 seconds for Zabbix. However, similar values, slightly higher than 50, can be observed for Lite and Small instances with both storage images.

A comparison between these results with the previous ones shown in Figure 6, highlights that the deployment time of VMs on INRIA is almost independent on the storage image and its dependence is higher with the instance type. This situation is remarkable for the DebSq10GV3 storage image. The main reason of this difference on the behaviour of deployment time between EPCC and INRIA is due to the configuration of the physical infrastructure. Both sites base its cloud infrastructure on the OpenNebula [10] platform but EPCC copies the storage images to the compute node using NFS. However, INRIA has implemented a system that makes a snapshot of the storage image after its first copy to the compute node via NFS. Therefore, one VM can be deployed faster on INRIA than EPCC if a snapshot of the storage image exists on the compute node.

Storage	Disk Size (MB)	Description
Zabbix	3400	Debian Squeeze with Zabbix monitoring
DebSqV3	604	Debian Squeeze
DebSq2GV3	2048	Debian Squeeze
DebSq10GV3	10240	Debian Squeeze

Table I
STORAGE RESOURCES AVAILABLE ON EPCC AND INRIA BONFIRE SITES.

Instance		vcpu	vmemory (MB)
EPCC	INRIA		
Lite	Lite	0.5	256
Small	Small	1	1024
Medium	Medium	2	2048
Large		2	4096
Xlarge		4	8192

Table II
INSTANCE TYPES AVAILABLE ON EPCC AND INRIA BONFIRE SITES.

Other difference, which rises from the comparison between sites, is the dependence with the instance type. Results on EPCC show that the deployment time is almost independent of the instance type however, medium instances on INRIA have larger deployment times. The origin of this difference is the way we are measuring the times, we do not have exclusive access to the infrastructure and the measured time also includes the scheduling time. Therefore, depending on the computational load of physical resources is possible to observe delays when instances with high computational requirements are requested.

These results rise several questions about the best way to design a quality of service model on federated clouds, since in some cases could not be possible to guarantee the same configuration among sites or the same overload of the infrastructure when users can choose the location of their experiments [11]. From a global point of view, a scheduling system with information about the load of the physical resources of each site belonging to the federated cloud may provide better allocation of the requested virtual machines. Furthermore, this scheduler might take into account other factors such as configuration differences of each site that can affect the deployment time of virtual resources [12].

 ISBN: 978-1-61208-216-5

IV. Conclusion and future work

This work described the experiment agent developed on the VCOC experiment supported by the BonFIRE project in order to facilitate the automatic deployment and monitoring of virtual clusters on the BonFIRE federated cloud. The experiment agent implements several functionalities such as accept the description of virtual clusters on a JSON file, error management, recording the deployment times of virtual resources or deployment of random experiments following a predefined pattern. The first use of the experiment agent was to study the deployment time of all possible combinations between the available storage images and instance types on EPCC and INRIA BonFIRE sites. The main objective of this study was to analyse the impact on the deployment time of allocating different requests. The obtained results show that the deployment time of a VM instance is around 50 seconds and 300 seconds depending on the size of the storage image, the instance type and the deployment type. These results were obtained without an exclusive access to the infrastructure and the measured time includes the scheduling time. The obtained differences between sites rise several questions about the best way to design a quality of service model on federated clouds. For instance, it is difficult to guarantee the same configuration among sites or the computational load of the physical infrastructure when users are able to choose the experiment location. The obtained results, together with other data from other experiments will help EPCC to modify their infrastructure in order to reduce deployment times for large images.

Acknowledgment

The work leading to this publication was funded by the European Commission's Seventh Framework Programme (FP7/2007-2013) under grant agreement no 257386, "BonFIRE". The authors would like to thank Mr. Kostas Kavoussanakis and Mr. Ally Hume from EPCC, and Mr. Florian Schreiner from Fraunhofer FOKUS, for fruitful discussions about the VCOC experiment. Finally, we must highlight that this work has been possible thanks to the technical support provided by Mr. Maxence Dunnewind from INRIA, Mrs. Eilidh Troup from EPCC, Mr. Roland Kübert from University of Stuttgart and Mr. Gareth Francis from EPCC.

References

[1] "BonFIRE project." [Online]. Available: http://www.bonfire-project.eu/ [retrieved: May, 2012]

[2] "VCOC: Virtual Clusters on Federated Cloud Sites ." [Online]. Available: http://www.bonfire-project.eu/innovation/virtual-clusters-on-federated-cloud-sites [retrieved: May, 2012]

[3] L. Carril, Z. Martín-Rodríguez, C. Mouriño, A. Gómez, R. Díaz, and C. Fernández, "Advanced Computing Services for Radiotherapy Treatment Planning," in *Cloud Computing*. CRC Press, Oct. 2011, pp. 529–551. [Online]. Available: http://dx.doi.org/10.1201/b11149-27

[4] "BonFIRE portal." [Online]. Available: http://www.bonfire-project.eu/innovation/virtual-clusters-on-federated-cloud-sites [retrieved: May, 2012]

[5] *Restfully*. [Online]. Available: https://github.com/crohr/restfully [retrieved: May, 2012]

[6] "JSON." [Online]. Available: http://www.json.org/ [retrieved: May, 2012]

[7] cURL. [Online]. Available: http://curl.haxx.se/ [retrieved: May, 2012]

[8] "httplib2." [Online]. Available: http://pypi.python.org/pypi/httplib2 [retrieved: May, 2012]

[9] "Finisterrare Supercomputer." [Online]. Available: https://www.cesga.es/en/infraestructuras/computacion/finisterrae [retrieved: May, 2012]

[10] *OpenNebula*. [Online]. Available: http://opennebula.org [retrieved: May, 2012]

[11] A. Andrzejak, D. Kondo, and S. Yi, "Decision Model for Cloud Computing under SLA Constraints," in *2010 IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*. IEEE, Aug. 2010, pp. 257–266.

[12] R. Buyya, S. K. Garg, and R. N. Calheiros, "SLA-oriented resource provisioning for cloud computing: Challenges, architecture, and solutions," in *2011 International Conference on Cloud and Service Computing*. IEEE, Dec. 2011, pp. 1–10.

 ISBN: 978-1-61208-216-5