PROCEEDINGS OF SCIENCE

Test of cloud federation in CHAIN-REDS project

Giuseppe Andronico[∗†]
Italian National Institute of Nuclear Physics, Division of Catania - Italy
E-mail: giuseppe.andronico@ct.infn.it

Roberto Barbera
Department of Physics and Astronomy of the University of Catania and INFN - Italy
E-mail: roberto.barbera@ct.infn.it

Salvatore Monforte
Italian National Institute of Nuclear Physics, Division of Catania - Italy
E-mail: salvatore.monforte@ct.infn.it

Maurizio Paone
Italian National Institute of Nuclear Physics, Division of Catania - Italy
E-mail: maurizio.paone@ct.infn.it

PoS(ISGC2014)013

Cloud Computing is considered a successful technology, involving distributed computation infrastructures, which take advantage of the virtualization of physical resources to actuate useful scale economies. In cloud ecosystems today, attention increasingly focuses on cooperation issues, mostly in the wider context of federation. Cloud federation refers to the process of interconnecting different cloud infrastructures for resource load balancing and to manage demand spikes. One of the biggest advantages of a federated cloud consists of the possibility of distributing the workload over different cloud providers, rather than obtaining cloud services from just a single supplier. In addition, federation could be very important for Big Data, for example, to implement policies allowing migration for virtual machines involved in data analysis next to storage hosting Big Data. Federation, automation, standards, and interoperability are crucial for cloud computing services to be successful in the near future. Standards are crucial for both interoperability and federation, and federation is the first step toward interoperability. The idea we describe in this paper comes from the assumption that some of Cloud Infrastructure offering IaaS (e.g., OpenStack, OpenNebula) provide interfaces based on the OCCI standard, but lack an interconnecting cooperation/federation awareness layer. We propose a software infrastructure able to provide a homogeneous interface among different Cloud managers, in order to coordinate the IaaS resources in a transparent and distributed manner. We have implemented such software, based on CLEVER, a novel Cloud Middleware jointly developed by INFN Catania and the University of Messina, which uses XMPP as interconnecting communication protocol. We also present:

- an interface for the Catania Science Gateway Framework implementing a simple dashboard
- a testbed used in the framework of the CHAIN-REDS Project, which includes sites belonging to the EGI Federated Cloud.

This work shows that cloud federation using standards is feasible.

International Symposium on Grids and Clouds (ISGC) 2014,
23-28 March 2014
Academia Sinica, Taipei, Taiwan

1. Introduction

Cloud federation generally refers to the process of interconnecting different cloud infrastructures to handle balancing the load that arises from users requests, yielding highly available services and satisfying the ever-growing demand of the involved Virtual Research Communities (VRCs).

A cloud federation should have two fundamental properties: transparency and elasticity. The former follows from the user's expectation to simply ask for resources, rather than to have to choose between available resources, while the latter comes from the assumption that a cloud infrastructure should be able to dynamically add more resources as required to meet user requests. Such a cloud federation is able to accommodate demand spikes transparently to the users and, at the same time, allows cloud administrators to implement policies aimed at optimizing the use of valuable resources, as required by the community of the CHAIN-REDS Project [1]. In this paper, we focus on two demonstrative use cases where cloud federation is the perfect choice to answer the requirements of the VRCs: *on demand application execution* and *elastic usage of resources available to a VRC*.

A generic illustration of our idea of cloud federation is depicted in Figure 1, which shows several independent cloud providers having made an agreement. When one of them, represented by the "Home Cloud", needs supplementary resources, it can elastically make use of resources made available from the other clouds participating in the agreement.

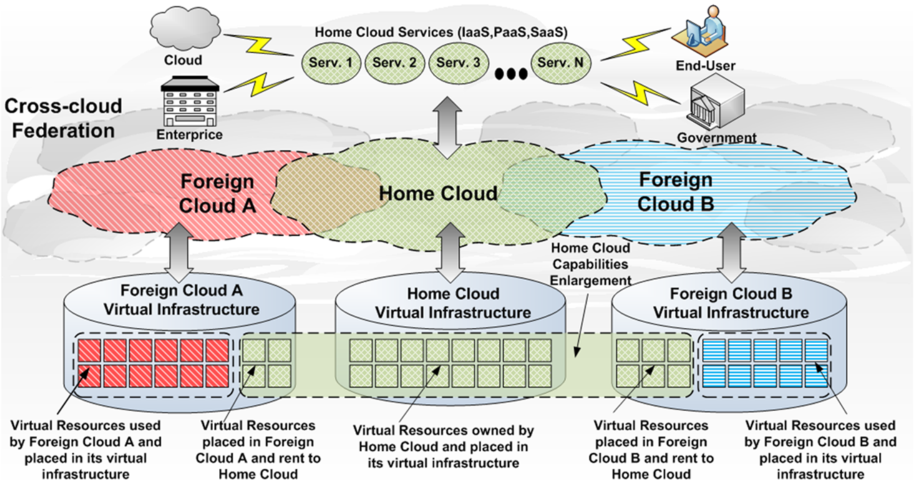

Home Cloud Services (IaaS,PaaS,SaaS)
Serv. 1
Serv. 2
Serv. 3
Serv. N
Cloud
Enterprice
End-User
Government
Cross-cloud Federation
Foreign Cloud A
Home Cloud
Foreign Cloud B
Home Cloud Capabilities Enlargement
Foreign Cloud A Virtual Infrastructure
Home Cloud Virtual Infrastructure
Foreign Cloud B Virtual Infrastructure
Virtual Resources used by Foreign Cloud A and placed in its virtual infrastructure
Virtual Resources placed in Foreign Cloud A and rent to Home Cloud
Virtual Resources owned by Home Cloud and placed in its virtual infrastructure
Virtual Resources placed in Foreign Cloud B and rent to Home Cloud
Virtual Resources used by Foreign Cloud B and placed in its virtual infrastructure

Figure 1: Generic cloud federation relationship

To meet the previously mentioned requirements, a Cloud Manager component built as a CLEVER [2] agent has been developed and integrated in a Liferay Portlet within the Catania Science Gateway Framework. (See [3] for a description).

2. CLEVER

CLEVER, designed and developed at the University of Messina, is novel Cloud middleware

[*] Speaker.

[†] CHAIN-REDS Project for economic support

for managing virtual appliances that provides an abstraction for the management of virtual resources and supports useful and easy management of private/hybrid clouds. It allows interactions of different "interconnected" computing infrastructures by handling simple and easily accessible interfaces.

The middleware is based on a distributed clustered architecture, where each cluster is organized as two hierarchical layers, as depicted in Figure 2. CLEVER nodes contain a host level management module, called Host Manager (HM). A single node may also include a cluster level management module, called Cluster Manager (CM).

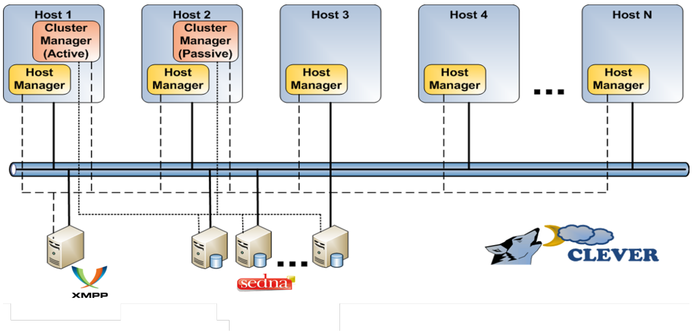


Figure 2: CLEVER elements

Both CMs and HMs are composed by several subcomponents, called agents, designed to perform specific tasks. Since agents are separate processes running on the same host (for fault tolerance), we based internal communication on Inter Process Communications (IPC), such as mechanisms exploiting D-Bus interfaces. On the other hand, the communication among CM, HMs, and external clients is based on the XMPP protocol [4], which was designed to drive communications in heterogeneous instant messaging systems, where it is possible to transmit any type of data. In particular, the protocol is able to guarantee connectivity among different users even with restrictive network security policies (NAT transversal, firewall policies, etc.). The XMPP protocol is also able to offer decentralized services, scalability in terms of number of hosts, flexibility in the system interoperability, and native security features based on the use of channel encryption and/or XML encryption.

To make CLEVER able to integrate different clouds, we added a new element (called Cloud Manager) to the Host Manager so that the new element will handle a remote cloud system similarly to how the HM handles a computing node.

As shown in Figure 3 the Cloud Manager is integrated in CLEVER, which allows CLEVER to integrate the resources provided by the remote cloud and the local resource pool.

OCCI (Open Cloud Computing Interface) allows standardization of the interaction between various cloud systems. In fact, the largest cloud systems provide an OCCI API to interact with and use the capabilities of the underlying middleware. Using this standard, heterogeneous remote resources can be joined together, enhancing system performance and availability. This standard, which provides a framework for the front-end infrastructure management of IaaS providers and is based on a REST (Representational State Transfer) approach, was developed by the OGF (Open Grid Forum) [5]. The Cloud Manager is a software agent for management of cloud environments that belong to different providers, allowing the administrator to coherently see all local and remote

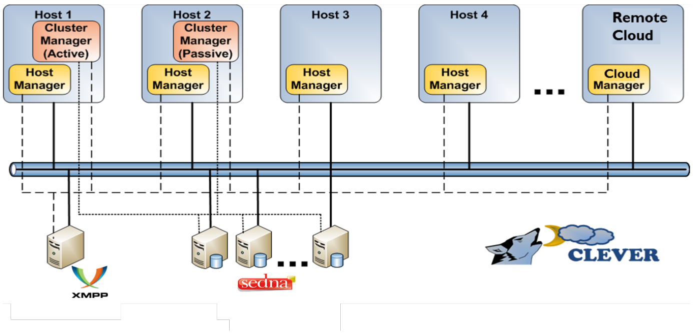


Figure 3: CLEVER elements with Cloud Manager

resources. CLEVER already natively contains the concept of federation between homogeneous clouds, and we have extended it to support heterogeneous environments through the OCCI standard. In this way, a user uniformly sees local and remote resources and is able to interact with the local or remote cloud, actually aggregating compute nodes from different types of cloud infrastructures.

3. Science Gateway

In current e-Science activities, Science Gateways are playing a steadily increasing role and their relevance will further increase with the development of more efficient network technologies and larger distributed computing infrastructures worldwide. Through the highly collaborative environment of a Science Gateway, users spread around the world and belonging to different organizations can easily cooperate to reach common goals and exploit all the resources they need to accomplish their work. A Science Gateway is defined as "community-developed set of tools, applications, and data that is integrated via a portal or a suite of applications, usually in a graphical user interface, that is further customized to meet the needs of a specific community" [6]. However, a Science Gateway is usually more than a collection of applications. Gateways often let users store, manage, catalogue, and share large data collections or rapidly evolving novel applications they cannot find elsewhere. Training and education are also a significant part of some science gateways.

Our Science Gateway is built within the Liferay web portal framework and portlet container [7] and it is fully compliant with the JSR 286 ("portlet2.0") standard [8]. Liferay is currently the most used framework to build Science Gateways in the "Grid world" and ships with more than sixty portlets that can be easily combined (mashed-up) to build complex and appealing e-collaboration environments. Other 200+ portlets are available in the repository of the Liferay community [9].

The highest level component in the authorization/authentication hierarchy has to be integrated in our Science Gateway and has to support a Single Sign On (SSO) mechanism across all services a given user is entitled (i.e., has the right) to use, in order not to confuse inexperienced users with different sets of credentials. Many web tools support SSO within a centralized or distributed authentication framework. Nevertheless, in order to comply with currently adopted standards and support the most relevant Identity Federations in Education and Research, the Science Gateway is compliant with the Security Assertion Markup Language (SAML) OASIS standard [10] for credential communication. The Shibboleth [11] implementation of SAML has been adopted in the

DECIDE Science Gateway [3] and a library has been developed to make Liferay manage the Shibboleth token (see Figure 4).

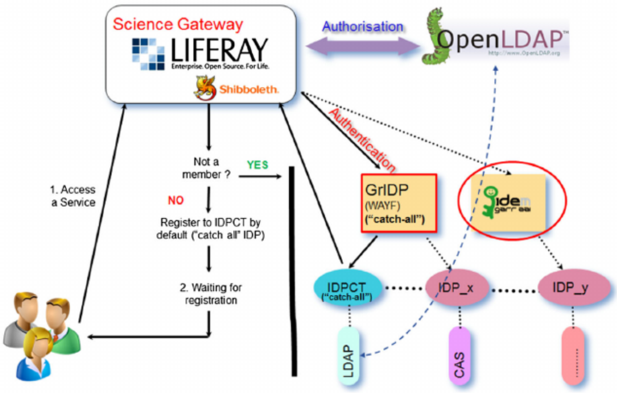


Figure 4: Reference Model for Science Gateway

Our Science Gateway uses the Catania Grid Engine to perform Grid transactions. The Catania Grid Engine is a generic software module able to interconnect the Scientific Gateway presentation layer with the underlying Grid infrastructures using standard technologies. It allows quick creation of new Science Gateways, providing their developers with a simple interface that lets them avoid worrying about middleware specificities. The adoption of international standards was considered from the beginning of the design of the DECIDE Science Gateway as a mandatory practice in order to protect the investment in the creation of this high-level user interface from middleware changes and lack of interoperability. In this way, the Scientific Gateway can become a "gate" to a huge e-Infrastructure made of resources coming from different kinds of Grid infrastructures with different kinds of middleware deployed and connected via standard interfaces. As a consequence of these considerations, the Catania Grid Engine adopts the Simple API for Grid Applications (SAGA) Core API [12, 13], a high level, application-oriented, software library for Grid applications specified by the Open Grid Forum (OGF), and its JSAGA implementation [14]. JSAGA allows the creation of a single interface to different middleware stacks and permits Science Gateways to exploit resources coming from different Grid infrastructures. A sketch of the Catania Grid Engine architecture is in Figure 5.

4. Use cases

Virtual Research Communities sometime need to run applications whose requirements do not simply fit grid or clusters frameworks. Cloud computing can be one possible solution since it simplifies the provisioning of specific Virtual Machines (VMs) on demand. In CHAIN-REDS, we provided a proof-of-concept solution that, transparently to the user, fulfills this requirement.

The workflow depicted in Figure 6 is composed through the Science Gateway Framework (SGF). Once the user is logged in, the user can access a list of user-authorization applications. When the user wants to submit a job, the SGF-based Science Gateway (SG) interacts with the cloud that offers the needed VM by means of OCCI. The required VM will start and the SG,

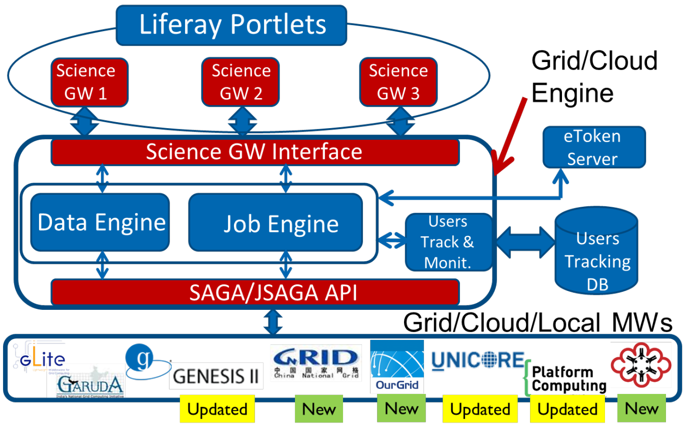


Figure 5: Science Gateway Framework components scheme

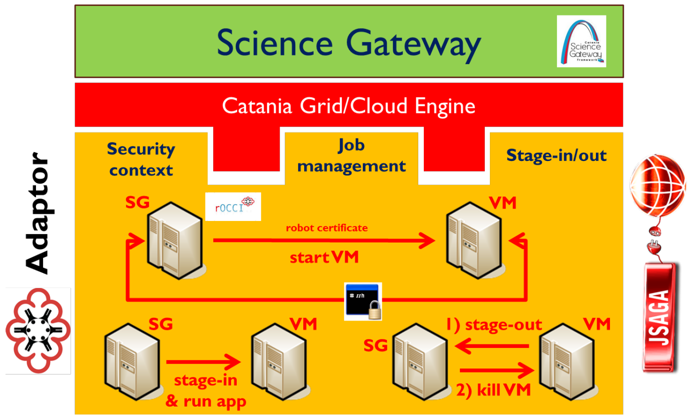


Figure 6: Workflow for use case 1

making use of JSAGA, submits the user job. Once the job ends, the SG uses again JSAGA to retrieve the output, makes it available to the user, and shuts down the VM.

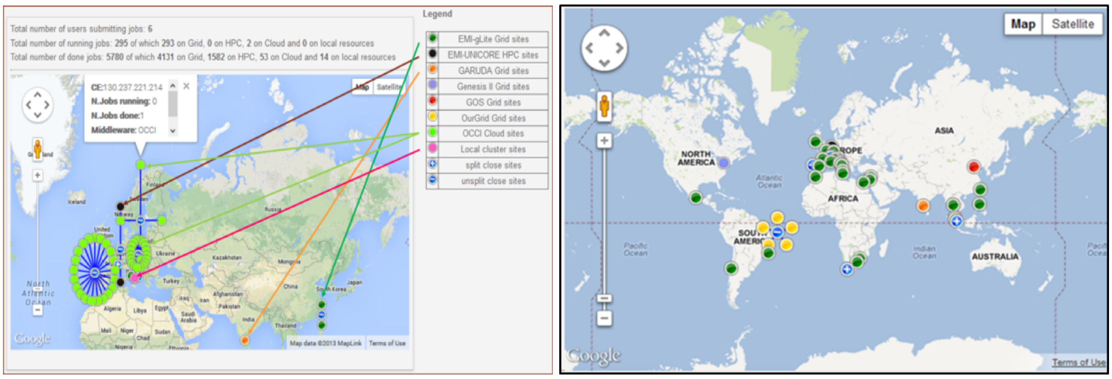
This feature is part of the CHAIN-REDS testbed. Figure 7a is a screenshot of the dynamic map showing the sites participating in the testbed in the European area. Involved partners provide resources in different ways: through clusters and grids, and via OCCI, through clouds. The Science Gateway supplies users with transparent and unified access to those resources.

Figure 7b shows the participating worldwide testbed sites.

Figure 8 summarizes the second use case introduced in Section 1, i.e., the efficient usage of resources made available to a VRC.

Eight different sites in six European countries have made part of their cloud resources available to produce a unified cloud, exploitable by users through a homogeneous interface. Three cloud manager stacks were involved among the cloud sites:

- Openstack [15]
- Openebula [16]
- Okeanos [17]



Total number of users submitting jobs: 6
Total number of running jobs: 295 of which 293 on Grid, 0 on HPC, 2 on Cloud and 0 on local resources
Total number of done jobs: 5780 of which 4131 on Grid, 1582 on HPC, 53 on Cloud and 14 on local resources

CE:130.237.221.214
N.Jobs running: 0
N.Jobs done:1
Middleware: OCCI

(a) Infrastructure in Europe for use case 1 (b) Sites for use case 1

Figure 7: Use case 1 screenshots

Some VM templates, instantiable and manageable by users, were defined and implemented to create a sort of marketplace. At this stage of development, we use a simple rsync-based procedure as a mechanism for image dissemination among partner sites (Figure 9). As shown in this figure, users can access the cloud resources of partner sites through a Web interface deployed as a portlet on the Science Gateway, called *MyCloud*. This portlet interacts with the various clouds through CLEVER, specifically with the Cloud Manager Agent, as described in Section 2.

A screenshot of this GUI is shown in Figure 10, where each box represents a remote cloud site, containing the VM instances created at that site by users. All the available template images are displayed on the right as a marketplace. Users can perform typical actions, such as VM (multi)deploy, start and stop, delete, etc. The IP assigned to each instance is published on DNS with *chain-project.eu* domain. Users can have SSH access to the instances directly via the GUI.

As depicted in Figure 11, both *application on demand* (the first use case) and *IaaS among heterogeneous clouds* (the second use case) have been deployed on Science Gateway infrastructure and can be accessed from the same web interface in an integrated environment.

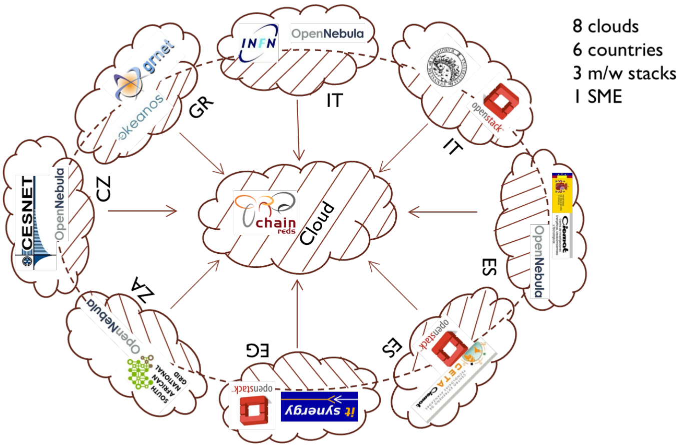


Figure 8: Testbed for use case 2

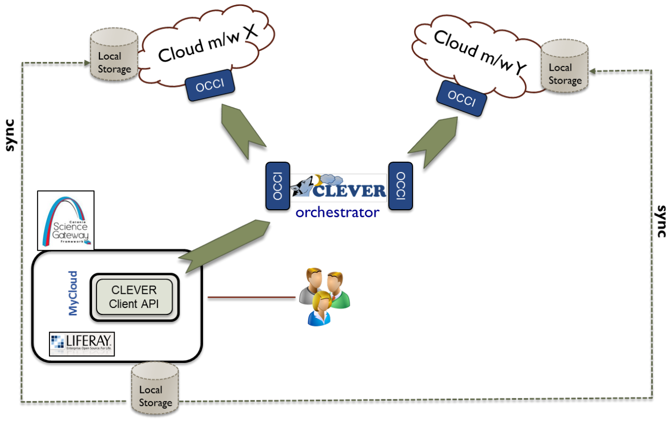


Figure 9: Logical scheme for use case 2

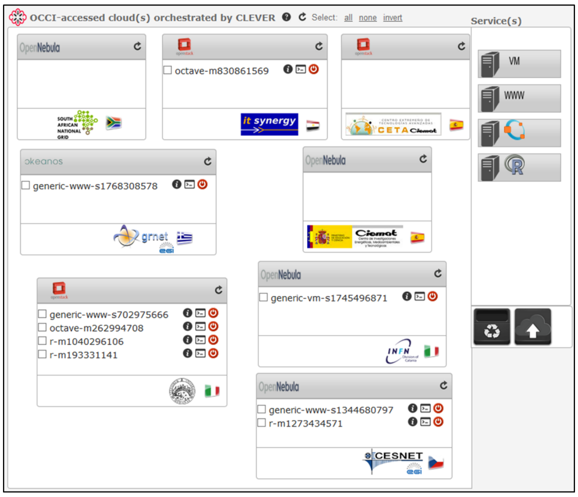


Figure 10: MyCloud GUI

5. Conclusions

In this paper, we have shown an integrated infrastructure able to provide both SaaS and IaaS capabilities, yielding a homogeneous interface to a set of heterogeneous cloud resources. In this scenario, two use cases have been presented, as well as their integration in the Science Gateway Framework.

The Grid Engine Framework and JSAGA adaptor have been described concerning the SaaS aspects, while Cloud Manager component developed on the CLEVER middleware have been analyzed in relation to the IaaS capabilities. The whole system exploits the OCCI standard to achieve a uniform interface among different cloud manager stacks.

References

[1] CHAIN-REDS. Chain-reds fp7 project. `http://www.chain-project.eu/`.

[2] Francesco Tusa, Maurizio Paone, Massimo Villari, and Antonio Puliafito. Clever: A cloud-enabled virtual environment. In *ISCC*, pages 477–482. IEEE, 2010.

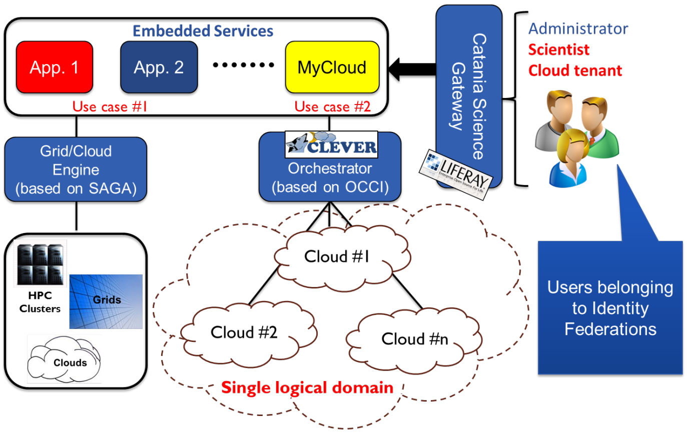


Figure 11: Scheme of the overall testbed

[3] V. Ardizzone, R. Barbera, A. Calanducci, M. Fargetta, E. Ingrà, I. Porro, G. La Rocca, S. Monforte, R. Ricceri, R. Rotondo, D. Scardaci, and A. Schenone. The decide science gateway. *Journal of Grid Computing*, 10(4):689–707, 2012.

[4] P. Saint-Andre. Extensible Messaging and Presence Protocol (XMPP): Core. RFC 6120 (Proposed Standard), March 2011.

[5] Open Grid Forum. An open community leading cloud. `http://occi-wg.org/`.

[6] Us teragrid project.

[7] Liferay. The liferay portal framework. `http://www.liferay.com`.

[8] JCP. The jsr286 standard. `http://www.jcp.org/en/jsr/detail?id=286`.

[9] Liferay. Liferay marketplace. `http://www.liferay.com/marketplace`.

[10] XML. The saml standard. `http://saml.xml.org`.

[11] Shibboleth Team. The shibboleth system. `http://shibboleth.net/`.

[12] Open Grid Forum. The saga ogf standard specification. `http://www.jcp.org/en/jsr/detail?id=286`.

[13] S. Jha, H. Kaiser, A. Merzky, and O. Weidner. Grid interoperability at the application level using saga. In *e-Science and Grid Computing, IEEE International Conference on*, pages 584–591, Dec 2007.

[14] JSaga team. The jsaga website. `http://grid.in2p3.fr/jsaga/`.

[15] Openstack Foundation. Openstack cloud manager. `https://www.openstack.org/`.

[16] OpenNebula. Opennebula cloud manager. `http://opennebula.org/`.

[17] GRNET. Okeanos cloud manager. `https://okeanos.grnet.gr/home/`.