Scientific Application Execution on Hybrid Platforms Based on Grid and Cloud Computing

G. Moltó, J.V. Carrión, V. Hernández
Instituto de Instrumentación para Imagen Molecular
Universidad Politécnica de Valencia
Camino de Vera S/N, 46022 Valencia, SPAIN
{gmolto,vhernand}@dsic.upv.es,jocarbur@upv.es

1. Introduction

The development of Grid computing technologies provided the scientific community with an appropriate infrastructure to share and to use computational and storage resources among geographically distributed organizations [1]. The recent advance in virtualization techniques has leveraged Cloud computing as a new source of computational and storage capabilities for applications. An Infrastructure as Service (IaaS) Cloud provides its users with an elastic platform, based on Virtual Machines (VM), that can grow and shrink according to the user's requirements and budget, since access is provided on a pay-per-use basis for public Clouds such as Amazon Elastic Compute Cloud [2].

The coordinated harnessing of Grid and Cloud resources is far from being a trivial task. While Grid computing leverages computational efficiency, the Cloud focuses on availability and scalability. This work aims at the collaborative usage of Grid and Cloud resources for the efficient execution of scientific applications. For that purpose we propose an architecture that enables to integrate the computational power of both computing paradigms via a system that orchestrates and abstracts the execution details with each infrastructure.

2. Proposed architecture and implementation

Figure 1 depicts the global architecture, which is divided in three main actors: The user, the Enactor and the computational resource infrastructures.

The interaction starts with the user who wants to execute a certain number of batch jobs. These jobs might have different computing requirements, both hardware (available RAM, temporary storage space, CPU, etc.) and software (dependences with third-party software, libraries, OS, etc.). For that purpose, a high level API (Application Programming Interface) and a modified JSDL (Job Submission Description Language) [3] can be employed for the description of the jobs and their requirements. The user submits the jobs to the Enactor which is responsible for the execution of the applications on the available Grid and Cloud infrastructures. This requires the development of efficient meta-scheduling techniques for hybrid Grid/Cloud infrastructures, and the development of gateways able to dispatch and monitor the jobs in the different infrastructures.

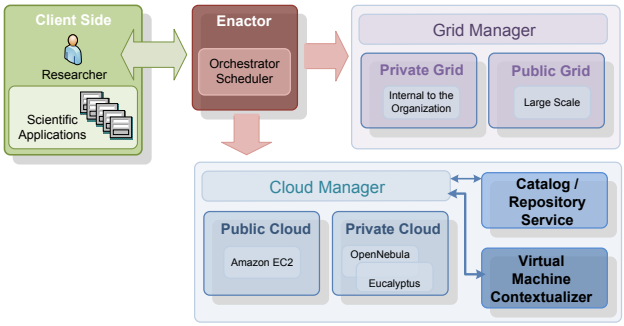

Figure 1. Platform architecture for the coordinated execution of scientific applications on Grid and Cloud infrastructures.

The goal of the Enactor is to reduce the global execution time of the jobs while optimizing the usage of the available resources under the budget constraints of the user. It is the central component of the architecture in charge of integrating resources from private Grids (typically available in research centers as development platforms) to public Grids (large-scale deployments such as the WLCG/EGEE Grid [4]) and from private Clouds (privately managed inside an organization) to public Clouds (such as Amazon EC2). Whilst Grids enable scientists to access a large pool of computing power, this is typically achieved by means of a research collaboration which requires specific credentials and has a limited lifetime. Instead, Cloud computing can be offered on a pay-per-use basis and the main advantage over Grids is that the resource provider no longer establishes the execution environment of the applications (as it happens with Grids). When no Grid resources are available, a Cloud provides the illusion of endless scalability and high availability. By using VMs, applications can be safely executed on a virtualized environment provided that they have all the hardware and software dependencies installed. This application contextualization process requires creating specific installation plans out of the job's requirements that guarantee the successful execution of the application on the virtual sandbox.

Therefore, in the case of Cloud infrastructures, the computational resources must be allocated on demand and they have to be properly contextualized (install all the software dependencies) to satisfy the job's requirements. For that, a catalogue and repository of Virtual Machine Images (VMI) is highly desirable in order to create generic VMIs that can be properly reused for different projects. For example, a VMI with the Matlab runtime installed and the Java JDK might be certainly reused for other applications with such dependences. This way, according to Fig. 1 the Enactor can contact the Cloud Manager to execute a certain job and this component uses the job's requirements to find the most appropriate VMI, which must satisfy all the physical requirements such as CPU and architecture and might possible satisfy part of the software dependences. Then, the VM Contextualizer module computes the deviation from the job's requirements to the state of this particular VMI in order to install the required software dependences into the VM.

3. Current prototype and next steps

We currently have a prototype implementation of the architecture based on our previous works on Grid and Cloud technologies. The GMarte Grid meta-scheduler [8] and its service-oriented counterpart [9] are being used to access Grid infrastructures and to coordinate the scientific application management and execution. Plugins to access Cloud infrastructures have been integrated in this software. In particular, we are using OpenNebula [5] for VM deployment but plans are to integrate other VM Managers such as Eucalyptus [6] and Nimbus [7]. A Virtual Machine Repository and Catalogue Service (VMRC) [10] has also been developed as the pivotal point of VMIs in the architecture. In addition, a prototype implementation of the contextualizer module has also been developed [11] which eases scientific application deployment via an XML-based declarative language to specify application installations. We have recently used this prototype for a Cloud-based application that aims at supporting the high availability of Grid services [12].

All these aforementioned works focus strictly on the Cloud side. In this particular work we aim at integrating Grid and Cloud resources to computationally support scientific applications. This requires efficient and cost-aware meta-scheduling techniques, that consider the economical (and energetic consumption) impact of starting new resources on the Cloud. This integration also poses the question of resource prioritization (when Cloud resources should be employed instead of Grid ones) in order to make the most effective usage of the infrastructures considering the jobs to be executed.

The coordinated usage of Grid and Cloud resources paves the way for new sources of computational power for resource-starved scientific applications. For that, it is important to integrate the different computational sources and to hide the infrastructure details in the shape of high-level tools that allow scientists to focus on job definitions instead of manual execution managements.

4. Acknowledgment

The authors would like to thank the financial support received from the Vicerrectorado de Investigación de la Universidad Politécnica de Valencia for the project PAID-06-09-2810 and to the Ministerio de Ciencia e Innovación for the project CodeCloud (TIN2010-17804).

5. References

[1] Foster, I, and C Kesselman. The GRID 2: Blueprint for a new computing infrastructure. Morgan Kaufmann, 2004.

[2] Amazon. "Amazon Elastic Compute Cloud (EC2)", 2010. http://aws.amazon.com/ec2.

[3] Anjomshoaa, A., F. Brisard, M. Drescher, D. Fellows, A. Ly, S. McGough, D. Pulsipher, and A. Savva. "Job Submission Description Language (JSDL) specification, version 1.0." In Open Grid Forum, GFD, 56, 2005.

[4] "WLCG: Worldwide LHC Computing Grid." http://lcg.web.cern.ch/LCG/public/.

[5] Sotomayor, Borja, Rubén S. Montero, Ignacio M. Llorente, and Ian Foster. "Virtual infrastructure management in private and hybrid clouds." IEEE Internet Computing 13, no. 5 (September 2009): 14-22.

[6] Nurmi, Daniel, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, and Dmitrii Zagorodnov. "The Eucalyptus Open-source Cloud-computing System." In Proceedings of 9th IEEE International Symposium on Cluster Computing and the Grid, 2009.

[7] Keahey, Kate, Renato Figueiredo, Jose Fortes, Tim Freeman, and Mauricio Tsugawa. "Science Clouds: Early Experiences in Cloud Computing for Scientific Applications.", 2008.

[8] Alonso, J M, V Hernández, and G Moltó. "GMarte: Grid middleware to abstract remote task execution." *Concurrency and Computation: Practice and Experience* 18, no. 15 (2006): 2021-2036.

[9] Moltó, G., V. Hernández, and J.M. Alonso. "A service-oriented WSRF-based architecture for metascheduling on computational Grids." *Future Generation Computer Systems* 24, no. 4 (2008): 317-328.

[10] Carrión, Jose V., Germán Moltó, Carlos De Alfonso, Miguel Caballer and Vicente Hernandez. "A Generic Catalog and Repository Service for Virtual Machine Images." In *2nd International ICST Conference on Cloud Computing (CloudComp 2010)*, 2010.

[11] Moltó, G, and V Hernández. "Management and Contextualization of Scientific Virtual Appliances." In *Cloud Futures 2010: Advancing Research with Cloud Computing*, 2010.

[12] Moltó, G., and V. Hernández. "On Demand Replication of WSRF-based Grid Services via Cloud Computing." In *9th International Meeting on High Performance Computing for Computational Science (VecPar 2010)*, 2010.