Cloud Computing Resource Management through a Grid Middleware: A Case Study with DIET and Eucalyptus

Eddy Caron, Frédéric Desprez, David Loureiro, Adrian Muresan

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Cloud Computing Resource Management through a Grid Middleware: A Case Study with DIET and Eucalyptus

Eddy Caron — Frédéric Desprez — David Loureiro — Adrian Muresan

N° 7096

October 2009

Domaine 3

ISSN 0249-6399 ISRN INRIA/RR--7096--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
GRENOBLE – RHÔNE-ALPES

Cloud Computing Resource Management through a Grid Middleware: A Case Study with DIET and Eucalyptus

Eddy Caron , Frédéric Desprez , David Loureiro , Adrian Muresan

Domaine : Réseaux, systèmes et services, calcul distribué
Équipe-Projet GRAAL

Rapport de recherche n° 7096 — October 2009 — 9 pages

Abstract: The Cloud phenomenon is quickly growing towards becoming the de facto standard of Internet Computing, storage and hosting both in industry and academia. The large scalability possibilities offered by Cloud platforms can be harnessed not only for services and applications hosting but also as a raw on-demand computing resource. This paper proposes the use of a Cloud system as a raw computational on-demand resource for a Grid middleware. We illustrate a proof of concept by considering the DIET-Solve Grid middleware and the EUCALYPTUS open-source Cloud platform.

Key-words: Cloud Computing, DIET-Solve, Eucalyptus, resource management

This text is also available as a research report of the Laboratoire de l'Informatique du Parallélisme `http://www.ens-lyon.fr/LIP`.

Centre de recherche INRIA Grenoble – Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Gestion des ressources d'une plate-forme Cloud grâce à un intergiciel de grille: Étude de cas avec DIET et Eucalyptus

Résumé : Le phénomène Cloud est en croissance rapide pour devenir le standard de facto du calcul, du stockage et de l'hébergement sur Internet que ce soit dans l'industrie ou dans les universités. Les possibilités d'extensibilité offertes par ces plates-formes peuvent être exploitées non seulement pour les services et l'hébergement d'applications, mais aussi en tant que ressource de calcul à la demande. Ce rapport de recherche propose l'utilisation d'un système de type Cloud comme fournisseur de ressources à la demande pour un middleware de grille. Nous illustrons cette preuve de concept en utilisant l'intergiciel DIET-Solve et la plate-forme open-source Eucalyptus.

Mots-clés : Cloud Computing, DIET-Solve, Eucalyptus, gestion des ressources

1 Introduction

Over the last years, Internet Computing and storage have considerably evolved from small isolated nodes to large-scale Cluster-like architectures driven by efficiency and scalability needs which we now know under the name of «Clouds» [3, 11]. They aim at being dynamically scalable and offer virtualized resources as service over the Internet. Usually solutions deployed in Clouds are web browser targeted and are balanced when it comes to computational power and storage. Clouds can also be used in more computational-intensive domains by using them as scalable computational resources.

Grid platforms offer a large benefit when it comes to computational power, yet they have a drawback caused by their scalability. To get the best of both worlds, a Grid middleware can be used to manage and harness raw Cloud computational resources.

The current paper will only scratch the surface of this interesting topic. We offer a proof of concept of using a Cloud system as computational resource through a Grid middleware. We demonstrate the use of EUCALYPTUS [9], the open-source Cloud, as a resource for the DIET [1] Grid middleware [5].

The rest of this paper is organized as follow. In Section 2, we present the overall architecture of EUCALYPTUS. Then we present the architecture of the Cloud version of DIET-Solve. After a short description of DIET-Solve, we give details about the way we connected it to EUCALYPTUS, the pro and cons of each approaches we tried and some performance estimations of the overheads. Finally and before a conclusion and future work section, we describe some previous works around this subject.

2 EUCALYPTUS

In the world of open-source Cloud Computing, EUCALYPTUS [8] is a pioneer. It relies on commonly-available Linux tools and simple Web Service technology and is compatible with the Amazon Elastic Computing Cloud (EC2) SOAP interface [2].

The EUCALYPTUS platform has a three-level hierarchical architecture. At the top of the hierarchy lies the Cloud Controller node (CLC). Its role is to coordinate the Cloud as a whole and to handle client Cloud management requests. This is the only node that is responsible for decision-making.

Halfway between the top and bottom of the hierarchy lies the Cluster Controller (CC). It is responsible for keeping track of resource usage in its Cluster.

At the bottom of the hierarchy lies the Node Controller (NC). Each physical machine that is to be a computing machine needs to have the NC service running. The NC has two main responsibilities: monitoring resource usage and managing virtual resources.

[1] DIET is now know under the name DIET-Solve.

To achieve virtual resource management, EUCALYPTUS uses Xen [2] and/or KVM [3] virtualization technologies. These offers great benefits when dealing with scalability and application isolation but have also drawbacks due to the necessary start-up and shutdown time of the virtual machines.

We have chosen to use the SOAP interface that EUCALYPTUS privides. This is more flexible than the query interface (the query string has a limited size) and offers better security because it implements the WS-Security standard with an asymmetric RAS key pair. As a result we have successfully manage to programmatically manipulate EUCALYPTUS virtual resources and gain direct access to them once instantiated.

On the basis of the above results we are ready to integrate EUCALYPTUS as a resource into DIET-Solve.

3 DIET-Solve over a Cloud

3.1 DIET-Solve Overview

The DIET-Solve component architecture is hierarchically structured for an improved scalability. The DIET-Solve toolkit [1, 5] is implemented in CORBA and thus benefits from the many standardized, stable services provided by freely-available and high performance CORBA implementations.

The DIET-Solve framework has several components. A **Client** is an application that uses the DIET-Solve infrastructure to solve problems using a GridRPC approach. A **SED** (**Server Daemon**) acts as the service provider, exporting functionality through a standardized computational service interface; a single SED can offer any number of computational services.

The third component of the DIET-Solve architecture, **agents**, facilitate the service location and invocation interactions of clients and SEDs. Collectively, a hierarchy of agents provides higher-level services such as scheduling and data management. These services are made scalable by distributing them across a hierarchy of agents composed of a single **Master Agent (MA)** and several **Local Agents (LA)**.

3.2 DIET-Solve Cloud Architecture

With the question of how to harness EUCALYPTUS as a DIET-Solve resource in mind we need to consider the architectures of both systems in order to find a suitable answer. From a high-level perspective we have several plausible solutions that differ by how much of the architectures of both systems overlap or are included one in the other. To be more precise we can consider the following two scenarios and any other scenario that is logically between the two.

DIET-Solve is completely outside of EUCALYPTUS: In this scenario the DIET-Solve and EUCALYPTUS architectures do not overlap at all in the sense that

[2] http://www.cl.cam.ac.uk/research/srg/netos/xen
[3] http://www.linux-kvm.org

all DIET-Solve agents or SEDs run separately with respect to the EUCALYPTUS controllers. The DIET-Solve SED requests resources (compute nodes) to EUCALYPTUS when needed and uses the resources directly (bypassing EUCALYPTUS broker) once they are ready. In this scenario scalability is limited because of the fixed DIET-Solve architecture that cannot scale easily, but the number of compute nodes takes full advantage of EUCALYPTUS's scalability.

DIET-Solve is completely included in EUCALYPTUS: The DIET-Solve architecture is virtualized inside EUCALYPTUS. This scenario is the other extreme of the previously stated one since DIET-Solve agents and SEDs are virtualized and instantiated on-demand. It is obvious that this scenario offers more flexibility because one can configure the amount of physical resources that are allocated to EUCALYPTUS virtual resources. This is also a more scalable approach because of the on-demand way of use that is typical to Cloud platforms.

The simplest and most natural scenario from the perspective of DIET-Solve is to treat EUCALYPTUS as a new type of resource allocator and scheduler inside the SED since DIET-Solve is easily extensible in this direction.

Figure 1 shows the architecture of DIET-Solve with three kinds of server daemons. **SED** The basic SED encapsulates a computational server. The information stored on a SED is a list of the data available on it, the list of problems that it can solve and performance-related information. **SED Batch** is an upgraded version of the previous one. This SED has the capability of submitting requests to different Batch Schedulers (BS) without the user having to know how to submit to the underlying BS. **SED Cloud** Based on the same idea that the previous one, this version provides the capability to use Cloud resources through the API of the Cloud platform. Knowing which image is associated to which service the SED could then automatically deploy it on the selected number of compute nodes.

Handling of a service call is done in three steps. **Obtain the requested virtual machines.** This first step involves requesting the virtual machines to EUCALYPTUS by using its SOAP API. The SED Cloud contains a mapping of services to virtual machines that the user is not aware of (and does not need to be aware of). The start-up of virtual machines is a time consuming operation and is best done asynchronously with the request. As a result, the SED Cloud polls EUCALYPTUS to receive a positive or negative reply related to the requested virtual machines. If the reply is positive then SED Cloud can proceed to the second step in handling the request. **Execute the MPI service on the instantiated virtual machines.** The end result of the previous step is a list of addresses of the instantiated virtual machines. Having this low-level information, SED Cloud can now initiate a MPI request on the machines. The result of the calculation is returned and a reply is formed for the service call. **Terminating the virtual machines.** The SED Cloud initiates another SOAP request to EUCALYPTUS for the termination of the instantiated virtual machines.

In order to test the functionality of the Eucalyptus scheduler and resource allocator, we set up a test architecture and deployed examples. The test architecture has two parts. **The DIET-Solve part**: a MA that published the services that DIET-

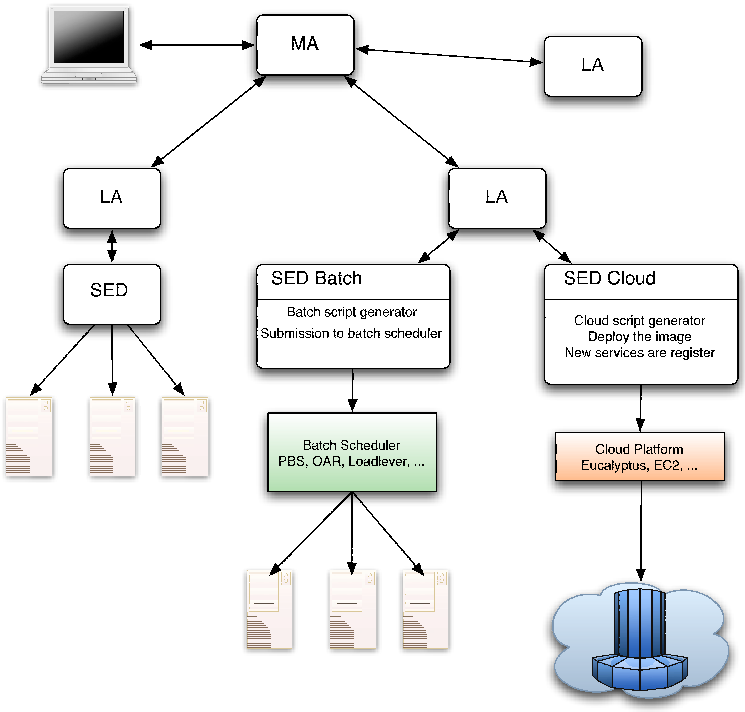

Figure 1: DIET architecture with three different kind of server daemons included Cloud server.

Solve makes available. The MA receives requests for service calls and relays them to the SED. The MA does not broker the request once the request has been relayed. A SED that is responsible for handling service call requests by instantiating EUCALYPTUS virtual machines. **The EUCALYPTUS part**: an EUCALYPTUS Cloud Controller (CLC) node that is the front of the cloud. All EUCALYPTUS requests and interactions start throught the CLC node. An EUCALYPTUS Cluster Controller (CC) node that manages the test Cluster. An EUCALYPTUS Node Controller (NC) installed on the physical machine that will be the compute node. Virtual machines will be instantiated on this node.

3.3 Performance Estimation

When refering to performance one is interested if the overhead outweigh the scalability increases and efficient use of resources. To get a good estimation we need to take into account the following performance factors: EUCALYPTUS instance throughput (a constant value that is added to the latency calculation and not a multiplication factor), EUCALYPTUS virtual network overhead and Xen virtualization overhead.

Instance startup time is the most difficult overhead to predict as it is influenced by the most varying factors like virtual machine OS and installed packages. However, we can get a good lower bound estimation for this value by considering a practical study done by UCSB Computer Science department [8]. By using “ttylinux" [10], a compact Linux image that boots very quickly, the startup times obtained by considering the number of requested instances are: 17 seconds for 1 instance and 24 seconds for 8 instances. We can consider these results as lower bounds when estimating instance throughput time.

To estimate the network throughput, the UCSB Computer Science department did performance network experiments and obtained the following average values: TCP throughput of 700 ms for 1 availability zone and 100 ms for 2 availability zones; round-trip latency of 1 ms for 1 availability zone and 5.5 ms for 2 availability zones.

For assessing Xen performance we refer to a study of virtualization systems for large scale emulations [4] that revealed that the CPU and memory overheads for Xen technology are negligible (by comparison to running in native mode). The study also revealed that these overheads are not influenced by the number of virtual machines run by the system, leading to a very good scaling.

In light of the above estimations we have optimistic expectations for the overall performance of the SED Cloud.

4 Related Work

In [6], the authors evaluate the use of Amazon EC2 for the coupling of ocean-atmosphere models. Both codes are written in MPI and are able to run on the

Amazon Clusters. Performance evaluation show the interest of such approach, even if dedicated supercomputers are still more efficient for such HPC applications.

The analysis of Clouds platforms presented in [7] is very interesting. Authors point out several issues that limit the use of Clouds for highly distributed applications. Among them, the level of abstraction at which Clouds are accessed is one of the major issues. Interoperability between different Cloud platforms is also an issue that will become a problem when their use will increase. DIET-Solve could be one of the approaches that will help GridRPC users to forget the underlying architecture on which they are running their programs.

5 Conclusion and Future Work

Cloud platforms have two advantages over their predecessors: a greater flexibility and an improved scalability. We propose the use of the DIET-Solve Grid middleware on top of the EUCALYPTUS Cloud system to demonstrate general purpose computing using Cloud platforms. We have seen possible ways of connecting the two architectures and the compatibility issues that need to be taken into account to achieve this. We have also described a working demo and performance estimations based on previous studies.

Although we have highlighted the most important issues that arise when dealing with this topic in this paper, our study serves solely as a proof of concept, leaving open questions as further development topics.

DIET-Solve schedulers must take into account the overhead of different SeDs when taking mapping decisions. We did not addressed the problem of data management. This is indeed an issue when running several request having data dependencies between them (for example when processing workflows).

References

[1] A. Amar, R. Bolze, Y. Caniou, E. Caron, B. Depardon, J. Gay, G. Le Mahec, and D. Loureiro. Tunable Scheduling in a GridRPC Framework. Concurrency & Computation: Practice & Experience, 20(9):1051–1069, 2008.

[2] Amazon Elastic Compute Cloud. http://aws.amazon.com/ec2/, 2008.

[3] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the Clouds: A Berkeley View on Cloud Computing. Technical Report UCB/EECS-2009-28, Electrical Engineering and Computer Sciences, University of California at Berkeley, 2009.

[4] Franck Cappello Benjamin Quetier, Vincent Neri. Selecting a virtualization system for grid/p2p large scale emulation. Technical report, 2006.

[5] E. Caron and F. Desprez. DIET: A Scalable Toolbox to Build Network Enabled Servers on the Grid. International Journal of High Performance Computing Applications, 20(3):335–352, 2006.

[6] C. Evangelinos and C.N. Hill. Cloud Computing for parallel Scientific HPC Applications: Feasibility of running Coupled Atmosphere-Ocean Climate Models on Amazon's EC2. In Cloud Computing and its Applications, Chicago, 2008.

[7] H. Jha, A. Merzky, and G. Fox. Using Clouds to Provide Grids with Higher Levels of Abstraction and Explicit Support for Usage Modes. Concurrency and Computation: Practice and Experience, 21(8):1087–1108, 2009.

[8] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. Eucalyptus: A Technical Report on an Elastic Utility Computing Architecture Linking Your Programs to Userful Systems. (2008-10), 2008.

[9] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Yousseff, and D. Zagorodnov. The Eucalyptus Open-source Cloud-computing System. In Cloud Computing and Its Applications (CCA-08), Chicago, October 22-23 2008.

[10] Ttylinux. http://minimalinux.org/ttylinux/.

[11] L.M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Linder. A Break in the Clouds: Towards a Cloud Definition. ACM SIGCOMM Computer Communication Review, 39(1):50–55, 2009.

Centre de recherche INRIA Grenoble – Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399