Architecture for the Next Generation System Management Tools for High Performance Computing Platforms

Geoffroy Vallée, Thomas J. Naughton, Anand Tikotekar, Jérôme Gallard, Stephen L. Scott, Christine Morin

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Architecture for the Next Generation System Management Tools for High Performance Computing Platforms

Geoffroy Vallée and Thomas Naughton and Anand Tikotekar and Jérôme Gallard and Stephen L. Scott and Christine Morin

N° 7062

October 2009

ISSN 0249-6399 ISRN INRIA/RR--7062--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
RENNES - BRETAGNE ATLANTIQUE

Architecture for the Next Generation System Management Tools for High Performance Computing Platforms

Geoffroy Vallée[*] and Thomas Naughton[*] and Anand Tikotekar[*]
and Jérôme Gallard[†] and Stephen L. Scott[*] and Christine Morin[†]

Thème : Calcul distribué et applications à très haute performance.
Équipe-Projet PARIS

Rapport de recherche n° 7062 — October 2009 — 11 pages

Abstract: Today, computational scientists mainly execute parallel or distributed applications, and try to scale up to get more results or greater data precision. As a result, they use more and more distributed resources, using local large-scale HPC systems (such as clusters or MPP), grids or even clouds. The difficulty of managing those platforms is their differences in nature, each degree abstracting some of the complexity created by resource distribution. For instance, clusters and MPP systems are located on a single site, composed of different "partitions" (*e.g.,* I/O nodes, compute nodes). In grids, "virtual organizations (VOs)" are one of the main concepts; since VOs are global and multi-users, they abstract both the complexity of the local resource management and account management away from the users. Finally, clouds provide an high degree of abstraction via the concept of "services", which can be implemented via a direct privileged access to the hardware or the usage of Internet based services. But all those cases require local management of resources and some kind of coordination (*e.g.,* coordination between partitions, remote sites, different administration domains).

This document presents a detailed description of the architecture of our novel system-management tool that can be used for the management of clusters/MPP systems, grids, and clouds. The architecture is based on three different concepts:

This work is done in the context of the INRIA SER-OS associated team – http://www.irisa.fr/paris/ser-os/.

[*] Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA – {valleegr, naughtont, tikotekaraa, scottsl}@ornl.gov – http://www.ornl.gov – ORNL research is sponsored by the Office of Advanced Scientific Computing Research; U.S. Department of Energy. The work was performed at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725.

[†] INRIA Rennes – Bretagne Atlantique, Rennes, France – firstname.lastname@inria.fr – The INRIA team carries out this research work in the framework of the XtreemOS project partially funded by the European Commission under contract #FP6-033576.

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu, 35042 Rennes Cedex
Téléphone : +33 2 99 84 71 00 — Télécopie : +33 2 99 84 71 71

(i) *Virtual System Environment (VSE)*, (ii) *Virtual Organizations (VOs)*, and (iii) *Virtual Platforms (VPs)*.

Key-words: Distributed Systems, Clusters, Grids, Clouds, Resource management.

Fondement pour la prochaine génération de systèmes de gestion des plates-formes de calculateurs à haute performance

Résumé : Les applications de calcul conçues pour être exécutées de manière parallèle ou distribuée peuvent généralement obtenir des résultats plus rapidement ou avec une plus grande précision en augmentant le nombre de ressources de calcul qui lui sont allouées. Il devient donc de plus en plus nécessaire d'utiliser, coinjointement ou non, des ressources distribuées hétérogènes, comme par exemple, des calculateurs à haute performance – HPC – (grappe de calculateurs ou MPP – architecture massivement parallèle), des grilles, des *Clouds*. À cause de la complixit'e et de la difference de nature des plates-formes hétérogènes, il devient de plus en plus difficile de les administrer et de les gérer de manière efficace. Par exemple, les grappes de calculateurs ainsi que les MPP sont situés généralement sur un seul site et sont partitionnés (par exemple, une partition pour les ressources de stockage, une autre pour les ressources de calcul). Dans les grilles, le concept "d'organisation virtuelle" (VO – *Virtual Organization*) permet d'abstraire la complexité de la gestion des ressources distribuées ainsi que celles des comptes utilisateurs. Enfin, les *Clouds* fournissent un haut degré d'abstraction en présentant la notion de "services". Ces services sont de différents niveaux en allant de l'accès privilégié aux ressources matérielles (virtuelles ou non) à l'utilisation de services de haut niveau proposés sur Internet.

Tout ces types de plates-formes de calcul nécessitent une gestion locale des ressources et une certaine forme de coordination (par exemple, la coordination entre différentes partitions, différents sites, ou encore différents domaines d'administration).

Ce document présente une description détaillée de l'architecture de notre système pour la gestion des nouvelles générations de plates-formes de calcul pour le HPC (grappes/MPP, grilles, *Clouds*). Ce travail est fondé sur trois différents concepts : (i) les environnements système virtuels *Virtual System Environment (VSE)*, (ii) les organisations virtuelles *Virtual Organizations (VO)*, (iii) les plates-formes virtuelles *Virtual Platforms (VP)*.

Mots-clés : Systèmes distribués, grappes de calcul, grilles de calcul, *Clouds*, gestion des ressources.

1 Introduction

Today, computational scientists mainly executing parallel or distributed applications, and try to scale up to get more results or greater data precision. As a result, they use more and more distributed resources, using local large-scale HPC systems (such as clusters or MPP), grids or even clouds.

Many system management tools have been studied and developed for those platforms but, to the best of our knowledge, none of the current solutions could be successfully used on all those platforms. We think that the best way to address challenges associated to the management of distributed resources is to focus on the *scientists' efficiency*: based on what the application scientists try to execute, the system should provide tools and mechanisms to abstract the distributed nature of computational resources, tolerate failures and provide the necessary runtimes.

For that, we propose an architecture based on three different concepts: (i) *Virtual System Environment (VSE)*, (ii) *Virtual Organizations (VOs)*, and (iii) *Virtual Platforms (VPs).* A VSE allows one to describe an application's needs in terms of software and can be deployed on any kind of single site / single administrative domain platforms (*e.g.,* using a disk-less or disk-full platform and/or real or virtual hardware). A VO allows one to deploy an application across multiple sites which can be in different administrative domains (VOs are global and multi-users).A VP is a view exposed to the users, based on virtual resources, that is local. This typically exposes the virtual hardware that locally fits the needs of a specific application. Based on VPs, it is possible to provide VOs, implementing system policies, typically for resource management (where should the VP be created?) and user access (who is allowed to access a given VP?).

Those three concepts have the benefit of allowing both the users and the system administrators to express their needs and constraints in term of execution environment: it is still necessary to provide system-level protection, system management and system monitoring capabilities. The key is to be able to "merge" scientists and system administrators needs, in order to get a single description of the execution environment which could then by deployed and used for application execution. Actually the system administrators can even choose the degree of isolation between the hardware and the application, based on the level of trust for users.

To ease the implementation of such a capability, it is useful to separate the configuration aspects and the actual deployment of the applications and their run time environments. This approach has the advantage of enabling negotiation for the configuration of a VP, *i.e.,* the negotiation between users and system administrators if conflicts are detected, and the negotiation between remote sites to get the best resource allocation. Additionally, the actual deployment of a VP is only performed when the configuration is complete, *i.e.,* when its configuration is available for everyone involved in the application execution. It also enables a fairly high security level since only configuration data are exchanged between entities involved in the application execution, and the VP being created locally. Note, the VP deployed locally could even be validated against the local system management policies, and the VP placed in a secure "sand box".

Therefore, this paper presents the architecture of a new system management tool that aims at enabling the management of clusters, MPP, grids, and clouds with the same infrastructure. For that, we present a systematic study of the requirements associated with the management of those platforms and, based on this classification, identify common requirements and capabilities, which ultimately leads to the presentation of a new system management tool. This tool is under development merging and extending existing tools: the OSCAR system management tool [7], the XtreemOS grid operating system [4, 6], and the Aladdin/G5K [2, 5] grid system management tools.

2 Terminology

Clouds, grids, clusters and MPPs are by nature different computing platforms: some span across multiple sites, and multiple administrative domains, whereas others are single site and single administrative domains.

Our goal is to federate tools for the management of such platforms. To accomplish this goal adequate abstractions are mandatory. Thus, before we describe more precisely the proposed system management software architecture for the maintenance of these diverse execution platforms, we describe the terminology used in the rest of this document. Many of the terms are already in use by the community, but we introduce the concept of *Virtual Platforms (VPs)* in order to link all the existing abstractions and make sure they can be used on all the target platforms.

The described abstractions aim to hide challenges created by the execution of parallel applications on distributed platforms, from the multi-site/multi-user case, down to the uni-site/uni-user case. Also note that throughout the document, a user is equivalent to an instance of an application, *i.e.,* we assume a user is running one and only one instance of a given application at a time. If a user wants to execute multiple applications, we assume those two applications are different since the associated requirements may be different. Table 1 gives an overview of our classification.

	Multi-site	**Single site**
Single user	Virtual Organizations (VO)	Virtual System Environments (VSE)
Multi-user		Virtual Platforms (VP)

Table 1: Classification of VOs, VPs, and VSEs

2.1 Virtual Organizations

A Virtual Organization (VO) [3] allows one to deploy an application across multiple sites which can be in different administrative domains, *i.e.,* VOs are *global and multi-users.* VOs are therefore well suited for the execution of applications across grids.

For that, *Virtual Breeding Environments (VBEs)* [8] are created on each site involved in a multi-site collaboration and can be used as a framework for the creation of a new VOs. In other words, VBEs span several domains of administration over different sites and are composed of a set of resources and users. A user can instantiate a new VO, based on a VBE, for which the new owner specifies the associated *roles*, *rules* and *resources*.

2.2 Virtual Platforms

A Virtual Platform (VP) is a local instantiation of a VO. It typically exposes the virtual hardware that fits the needs of a specific application on a given platform. Restated, a set of *virtual resources* is put together to create a VP, which is a subset of the local hardware resource. Doing so, it is possible to partition the local resources and allocate them to different VOs. Therefore at a given site, it is possible to instantiate VOs, "partitioning" the system. The local instantiation of the VO implements the associated system policies, typically for resource management (*e.g.,* where the VP should be created?), and for user access (*e.g.,* who is allowed to access a given VP?).

A Virtual Platform (VP) is a view exposed to the users, based on virtual resources, *i.e., local and multi-user.* A VP is then associated to a VSE for its actual software configuration.

2.3 Virtual System Environments

A Virtual System Environment (VSE) allows one to describe an application's software needs and can be deployed on any kind of single site / single administrative domain (*e.g.,* using a disk-less or disk-full platform and/or real or virtual hardware) [1].

The software requirements for a given application are typically constraints on the operating system (OS) and run-time environment (RTE). For instance, an MPI application can be designed to run on top of Red Hat Enterprise Linux 4.0 with LAM/MPI 7.1.3. If those constraints change (*e.g.,* update of the target platform software configuration), most likely the application will have to be modified, *ported.* Furthermore, it is important to decouple the definition of the application's needs in terms of RTE and what components system administrators want to have in each environment used by applications.

However, the science resides in the applications and not in the technical details about the requirements for the execution of those applications on HPC systems. In other words, application developers should not have to deal with application modifications due to (undesired) general software updates, which do not fit their scientific roadmap.

System-level virtualization enables the creation of *virtual appliances*, *i.e.,* virtual machine images customized for the execution of a given application. However, the definition of the application environments in such appliances is still mainly manual. Furthermore, because of a lack of genericness (the concept of appliance is deeply tight to system-level virtualization), it is not possible to deploy an existing virtual appliance on top of a standard system (*i.e.,* disk-less or disk-full systems).

The VSE is therefore a meta-description of the run-time environment required by a given application for a single site in a single domain of administration, and a single user.

3 Management of Execution Environments on Distributed and Parallel Platforms

Figure 1 presents the proposed software architecture for the management of clouds, grids, clusters and MPPs. This management software is primarily composed of two parts: i) the tools used to describe the configuration for the application's execution environment, and ii) the tools that will actually deploy this environment on a set of resources.

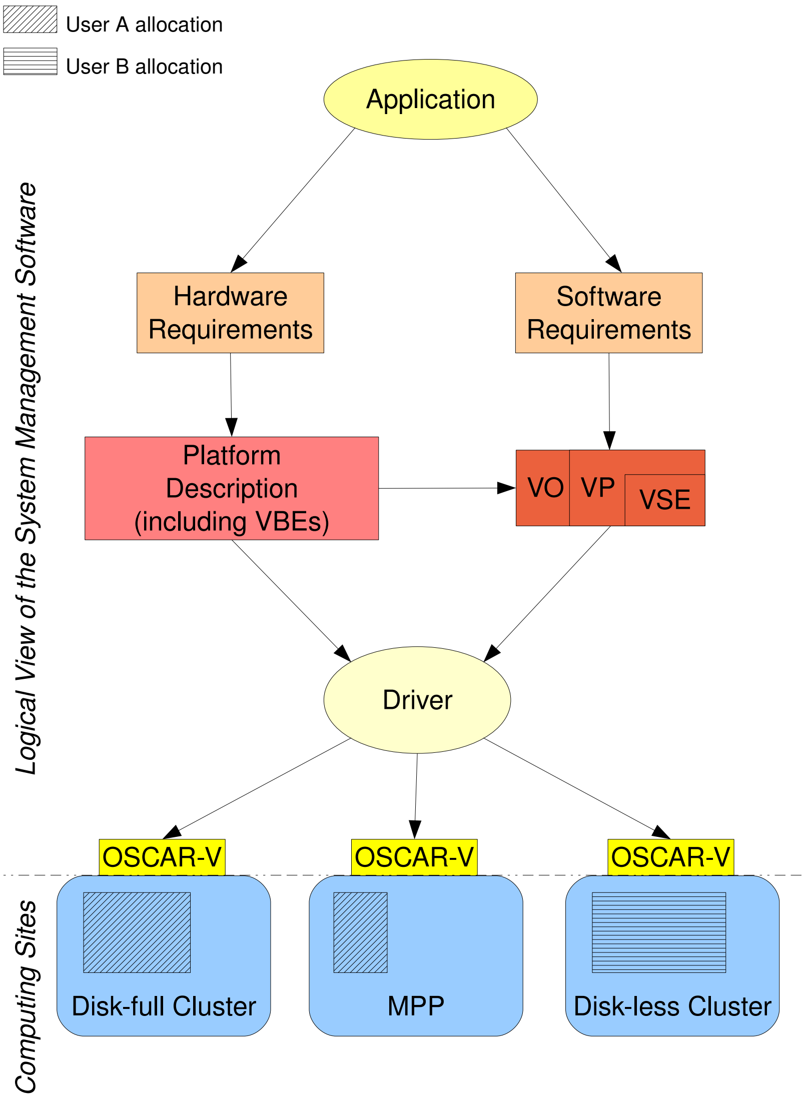

Figure 1: Overview of the System Management Software

3.1 Configuration of the Application Execution Environment

To prepare the application execution environment, the proposed software is based on the application description (both in terms of software and hardware), and on the description of available resources.

The description of available resources is based on *VBEs*. A VBE is composed of a set of resources (*e.g.,* computers, users) and a set of policies that specifying what VBE members can do. When an application is submitted, a new VO is created based on a VBE.

Based on this configuration data, it is possible to define a VO and how to deploy it on the target platforms. On each site, this deployment is based on a VP which is itself based on a VSE: each site only receives the description of the environment that needs to be deployed locally (*i.e.,* a VP is the local instantiation of a VO), based on a software environment (specified by a VSE).

This approach allows us to only exchange configuration data, limiting the amount of data that needs to be exchanged between the different sites. This also allows us to send each site configuration data that is specific to the local configuration: data about the global configuration is not sent to each node, increasing the overall security of the system.

3.2 Deployment of Application Execution Environments

Once the overall description of the environment needed for the execution of a given application is determined, it is possible to instantiate that environment on the target nodes. Since those nodes can span multiple sites and multiple administrative domains, a driver is in charge of selecting the appropriate sites and sending the configuration data specific to those sites.

At each site, OSCAR-V [9] actually receives the configuration information from the driver and ultimately deploys the environment needed for the execution of the application. By leveraging OSCAR-V, this environment can be based on a number of system configurations, to include: standard disk-full & disk-less system configurations and physical & virtual machines.

3.3 Implementation

The XtreemOS prototype currently provides tools for the deployment and management of VOs based on the concept of a VBE. In contrast, OSCAR-V currently supports the concept of a VSE, and can be used for any combination of disk-less/disk-full systems and physical/virtual hardware. The concept of a virtual platform (VP) is currently under development and will be a central piece for the integration of OSCAR-V and XtreemOS.

4 Conclusion

In this paper, we presented the architecture of a new tool for the management of clusters, MPPs, grids and clouds. The key of the architecture is to include adequate abstractions so many software “components” could be reused.

Furthermore, one of the major benefits of the proposed architecture is to adapt the system environment to user needs without compromising the control capabilities of system administrators. This enables scientists to easily describe their application's needs in terms of software and the hardware resources and then the system will automatically create the correct environment for the target systems. For that, the proposed architecture includes abstractions for standard resource, users and application management software.

Ultimately, the proposed architecture provides powerful tools that could be reused in many different contexts, guaranteeing stability and compatibility which ultimately should ease the scientists' life.

Contents

References

[1] Virtual system environments. In *Systems and Virtualization Management. Standards and New Technologies*, volume 18 of *Communications in Computer and Information Science*, pages 72–83. Springer Berlin Heidelberg, October 21-22, 2008.

[2] Raphael Bolze, Franck Cappello, Eddy Caron, Michel Daydé, Frederic Desprez, Emmanuel Jeannot, Yvon Jégou, Stéphane Lantéri, Julien Leduc, Nouredine Melab, Guillaume Mornet, Raymond Namyst, Pascale Primet, Benjamin Quetier, Olivier Richard, El-Ghazali Talbi, and Iréa Touche. Grid'5000: a large scale and highly reconfigurable experimental Grid testbed. *International Journal of High Performance Computing Applications*, 20(4):481–494, November 2006.

[3] Massimo Coppola, Yvon Jégou, Brian Matthews, Christine Morin, Luis Pablo Prieto, Óscar David Sánchez, Erica Y. Yang, and Haiyan Yu. Virtual organization support within a grid-wide operating system. *IEEE Internet Computing*, 12(2):20–28, 2008.

[4] Toni Cortes, Carsten Franke, Yvon Jégou, Thilo Kielmann, Domenico Laforenz, Brian Matthews, Christine Morin, Luis Pablo Prieto, and Alexander Reinefeld. XtreemOS: a Vision for a Grid Operating System. Technical report 4, XtreemOS European Integrated Projet, May 2008.

[5] Jérôme Gallard, Oana Goga, Adrien Lebre, and Christine Morin. VMdeploy, Improving Best-Effort Job Management in Grid5000. Technical Report RR-6764, INRIA, December 2008.

[6] Christine Morin. XtreemOS: A Grid Operating System Making your Computer Ready for Participating in Virtual Organizations. In *ISORC'07: Proceedings of the 10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing*, pages 393–402, Washington, DC, USA, 2007. IEEE Computer Society.

[7] John Mugler, Thomas Naughton, Stephen L. Scott, Brian Barrett, Andrew Lumsdaine, Jeffrey M. Squyres, BenoÃ®t des Ligneris, Francis Giraldeau, and Chokchai Leangsuksun. OSCAR Clusters. In *Proceedings of the* 5^{th} *Annual Ottawa Linux Symposium (OLS'03)*, Ottawa, Canada, July 23-26, 2003.

[8] NathalíÂe Galeano Sánchez, David Apolinar, Guerra Zubiaga, Jaime Atahualpa, Irigoyen González, and Arturo Molina. Virtual Breeding Environment: A First Approach to Understanding Working and Sharing Principles. 2008.

[9] Geoffroy Vallée, Thomas Naughton, and Stephen L. Scott. System management software for virtual environments. In *CF '07: Proceedings of the 4th international conference on Computing frontiers*, pages 153–160, New York, NY, USA, May 7-9, 2007. ACM.

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
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 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