Integrating Heterogeneous Grid Computing Environments

Alessandra C. G. Nascimento[1], Marcio N. Miranda[3], Fábio M. Costa[2]

[1]Faculdade de Goiás (FAGO)
Rua 67-A 216 – 74063-321 – Goiânia, GO – Brazil

[2]Instituto de Informática – Universidade Federal de Goiás (UFG)
Caixa Postal 131 – 74001-970 – Goiânia, GO – Brazil

[3]Instituto de Computação – Universidade Federal de Alagoas (UFAL)
BR-104 Norte km 97 – 57072-970 – Maceió, AL – Brazil

alessandragarcia2004@yahoo.com, fmc@inf.ufg.br, marcio@tci.ufal.br

Abstract. *Grid computing provides integration of different environments, creating a unified system, where organizations can share, manage and access different resources regardless of where they are located. However, the lack of integration among different grid computing systems hinders total exploitation of resources. This work investigates and develops an interface for the integration of two heterogeneous grid computing environments: Globus and InteGrade. Such integration transparently increases computational power. Users will be able to submit jobs through both grid environments and applications will be executed in either Globus or InteGrade grids, or even in both.*

Introduction

Grid computing [Foster 2001] systems aim to integrate idle computing resources and are becoming widely used. Grids are no longer a simple method of aggregating widely distributed resources, with a single focus on the high performance computing community (high performance grids): they have became an infrastructure for the integration of systems on a service-oriented basis (services grid). Several tools have been designed to implement grid computing application platforms. We can mention the Globus Toolkit [Globus 2006], InteGrade [Integrade 2006], Condor [Condor 2005] and OurGrid [Ourgrid 2005], among others.

Although conventional technologies that compose the current Internet allow communication and exchange of information between computers, they do not allow an integrated approach for shared and coordinated use of resources belonging to different institutions in the accomplishment of more complex computations. In this scenario, it is common to find a large amount of idle resources co-existing with an increasing demand for computational capacity. The demands imposed by high-performance applications (such as those related to the areas of the health, genetic and physics of high energy) could be supplied through the exploitation of these idle resources. The main objective of grid computing is to provide an infrastructure to exploit these idle resources. Grid Computing introduces the concept of virtual computational resources [Foster 2004], which is a basic requirement for computing on demand (use of available idle computational resources), also allowing the creation of virtual organizations around thematic projects based on distributed environments.

This work proposes an approach for the integration of heterogeneous grid computing environments, which combines environments that work with OGSA architecture (Open Grid Services Architecture) [OGSA 2005] and environments that work with CORBA technology [OMG 2003]. We aim to be able to run applications on a heterogeneous grid computing environment in a transparent way, without worrying about changes in source code and ways of allocating computational resources.

Grid Computing Environments

One of the main characteristics of a grid is to deal with the heterogeneous equipments that compose it. Grids must be capable of communicating with each other. This integration is a challenge if we consider that different grids are controlled by different and independent organizations, which do not share the same administration, the same politics or even the same tools. A common scenario is to have different grid middleware solutions that need to be integrated in order to fully utilize the resources of the distributed environment.

Globus Toolkit

Globus Toolkit [Foster 2001] is a set of tools and libraries to support the grid architecture and applications. It is a project developed by the Argone National Laboratory (ANL) and by the University of Southern California. The main services of the infrastructure are: the Globus Resource Allocation Manager (GRAM), the Monitoring and Discovery Service (MDS), Reliable File Transfer (RFT) and the Grid Security Infrastructure (GSI). Globus Toolkit 4 (GT4) [Globus 2006] is based on the concept of Grid Services, an expansion of the concept of Web Services. The main concepts and standards on which GT4 is based are: OGSA, OGSI (Open Grid Software Infrastructure) [OGSI 2005] and Web Services.

InteGrade

InteGrade [Finger 2003] is a grid middleware developed by The University of São Paulo to allow the implementation of grids over non-dedicated computational resources. The main characteristics of InteGrade include: objected-oriented architecture, the use of CORBA as communication infrastructure and support to non-trivial parallel applications. It also collects machine usage patterns in order to improve scheduling. The basic structural unit of InteGrade is the cluster. A cluster is a set of machines (nodes) grouped according to a specified criterion. A *dedicated node* is a machine that is reserved for exclusive use through the grid. *Shared nodes* are the ones belonging to individual users, who shares their idle resources with the grid. Finally, the cluster manager is the node that run the modules in charge of information collection and task scheduling.

Integration Architecture

The developed methodology uses, as a case study, computational grid environments already deployed, as well as simulations in real environments. With this purpose, the tools Globus and InteGrade are used. The most important services in the integration of the Globus and InteGrade environments are: security, remote submission of tasks and management of available information and resources in the grid. These are essential services to keep a grid working. Figure 1 presents the integration architecture between Globus and InteGrade systems. The user submits the application to the grid in a transparent way without worrying about which type of system will control and manage the available resources for this execution. Applications are submitted through Globus, using command "globus-run" or through InteGrade using the ASCT tool. The applications scheduler will decide how to distribute the application in the grid according to the parameters given by the user. Globus and InteGrade do not exchange information because they have different architectures and protocols. To solve this problem, an adapter has been created in order to work like a proxy, transfering applications from one side of the grid to the other. Information about the available resources in the grid can be obtained using the MDS component of the Globus.

Globus and InteGrade integration is divided in 2 parts, according to Figure 2. In the first part, the Globus user submits tasks to be processed by the InteGrade grid. Globus considers the InteGrade grid as a local scheduler, where GRM receives requests for processing tasks in its LRM nodes. The Globus GRAM has a set of modules that implement interfaces for the local schedulers of computational resources. These interfaces are in charge of managing task submissions to the scheduler. GRAM has interfaces for several types of local schedulers that belongs to other grids. This flexibility to accept new local schedulers, using the set of existing interface modules, allows construction of an interface for the InteGrade GRM.

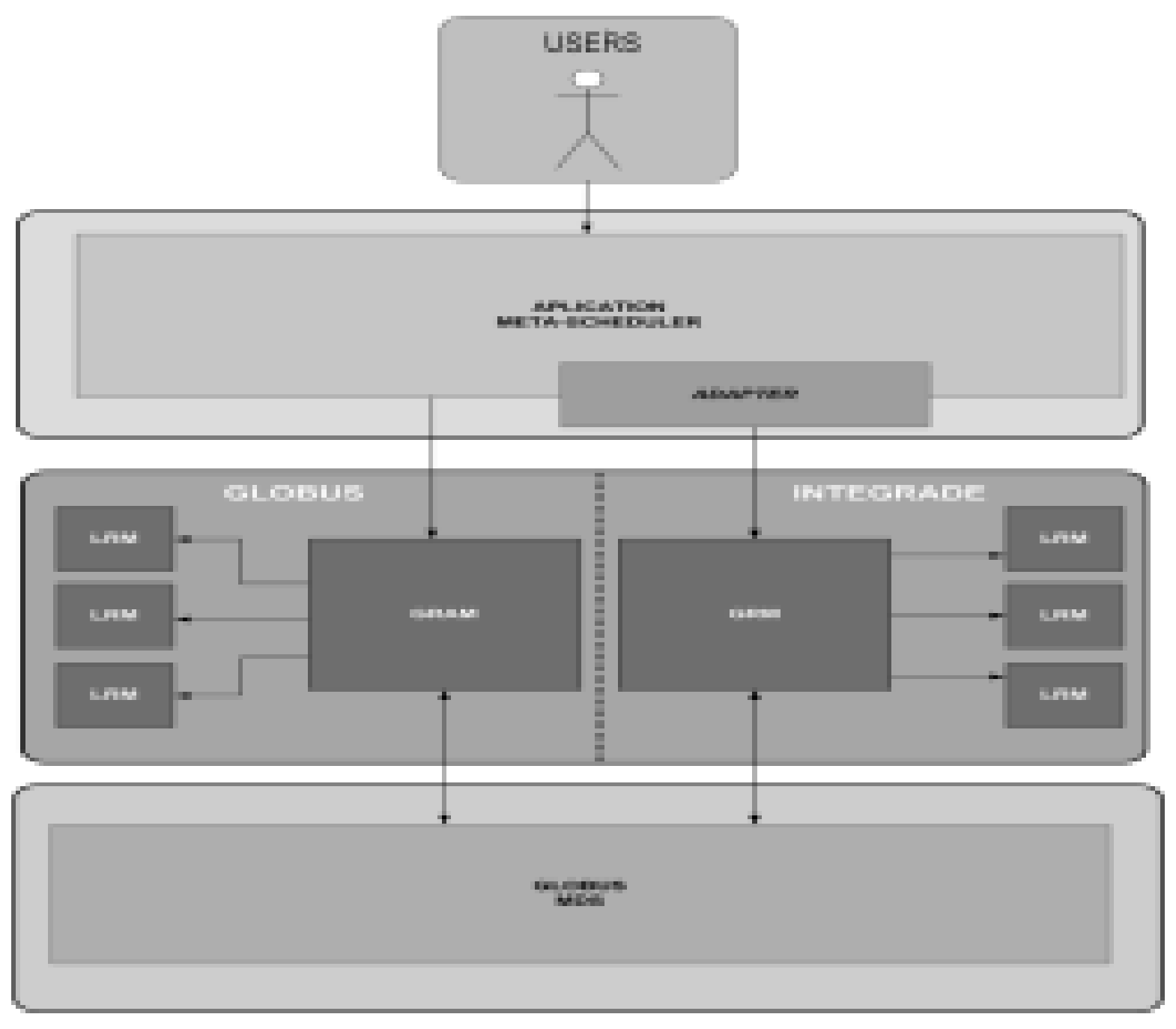

Figure 1 – Integration architecture

Figure 3 shows the Integrade architecture as an external scheduler of the Globus grid. Two new components have been inserted in the Globus Toolkit architecture in order to make possible to the InteGrade to process tasks sent by Globus: the InteGrade adapter, a module written in the Perl language, which uses the Globus interface, and the SGE which allows the monitoring of tasks sent from InteGrade to the Globus.

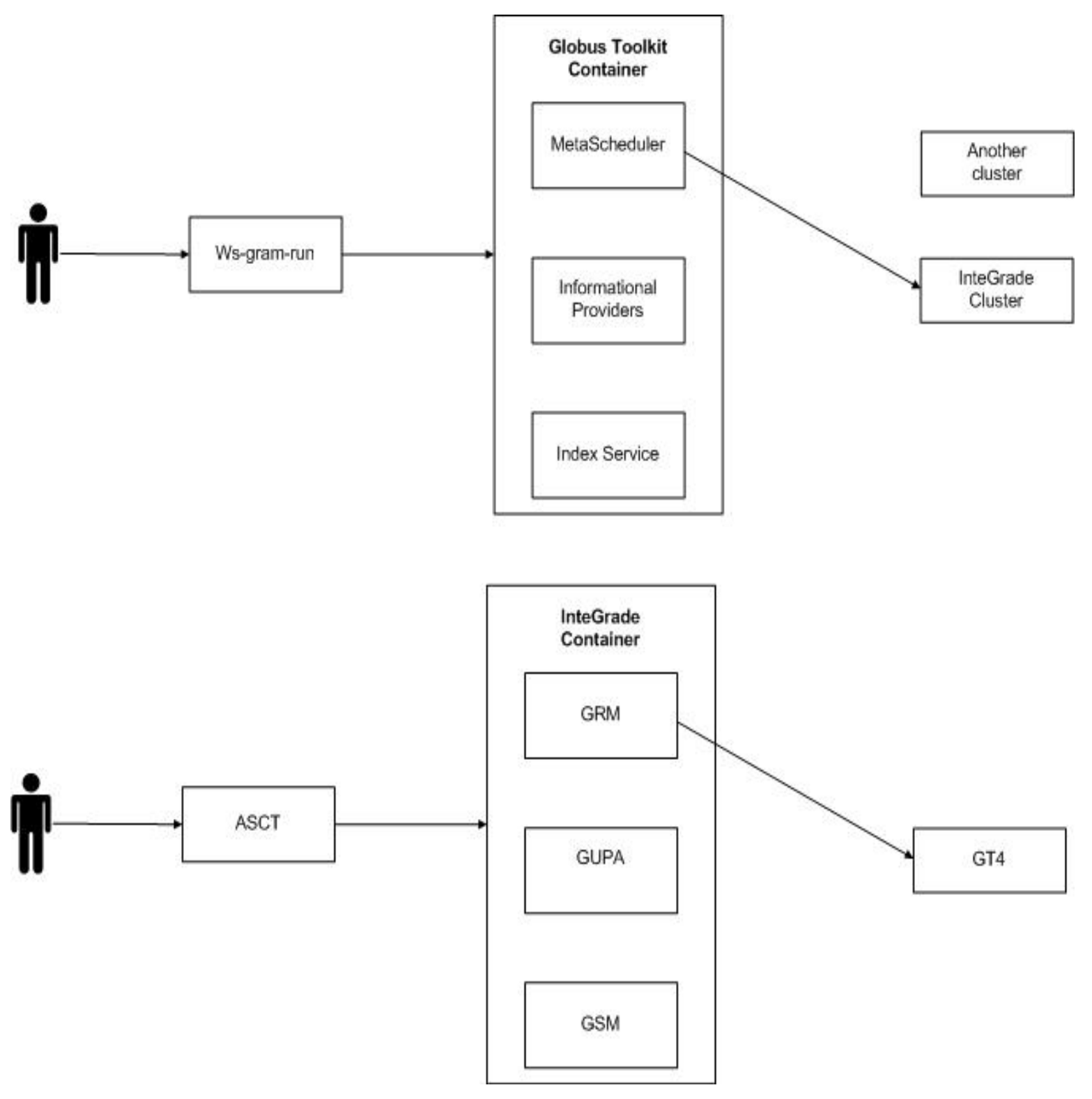

Figure 2 - Integration Models

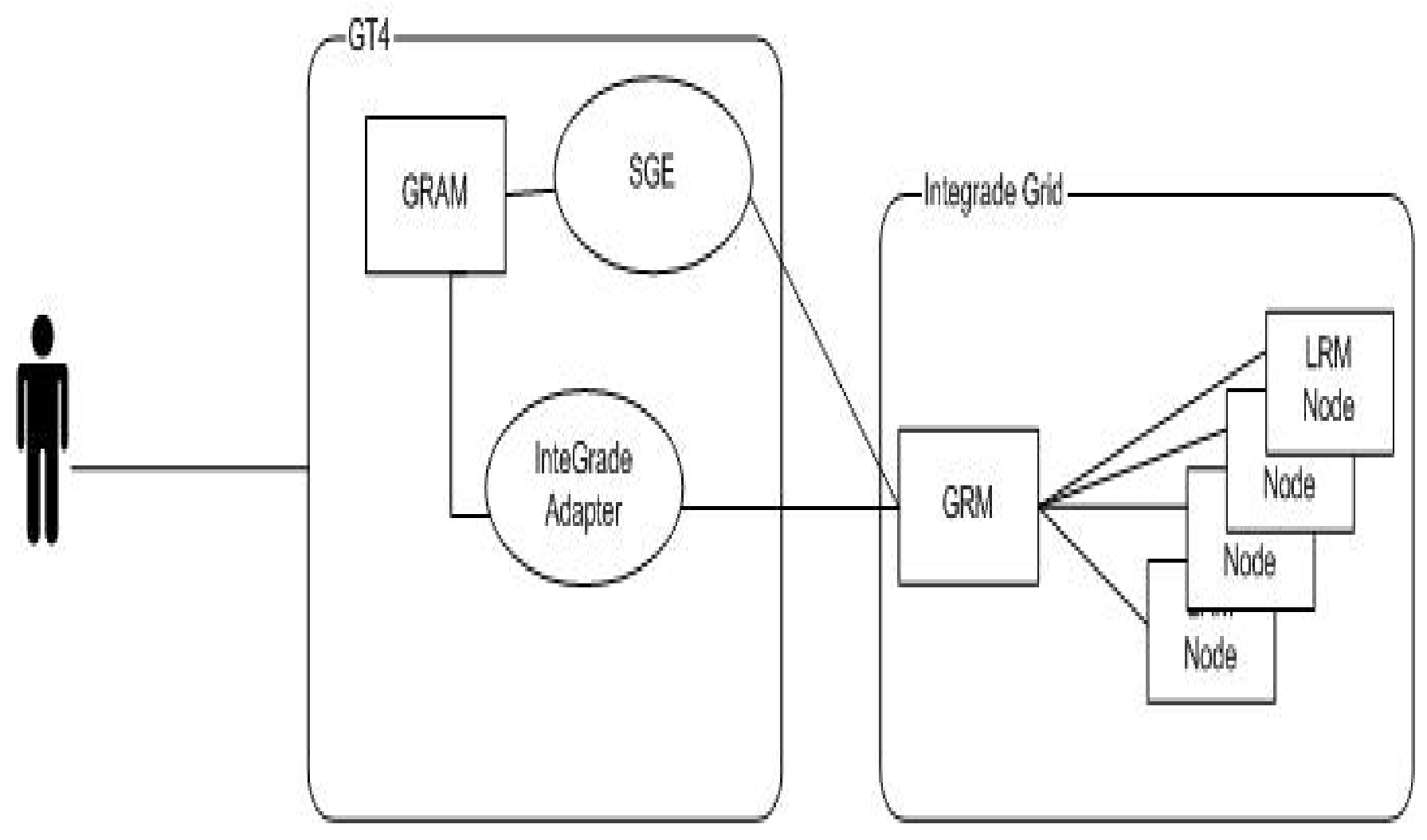

Figure 3 - InteGrade like external scheduler

Figure 4 illustrates how an InteGrade user submits tasks to be processed in Globus (GT4). For the scheduling of available computational resources, the Integrade GRM does not consider only its LRM nodes because Globus will be seen as an InteGrade LRM node, capable to receive tasks to be executed. With this purpose, a new component in InteGrade is created, called LRM Globus Node, which works like a proxy. Its function is to translate InteGrade information into Globus information. This new node also supplies information about the GT4 to the GRM. This kind of integration between the two systems avoids changes in the architecture or in the code of its components. Moreover, a version of ASCT was created in order to provide a command line interface to integrate with Perl.

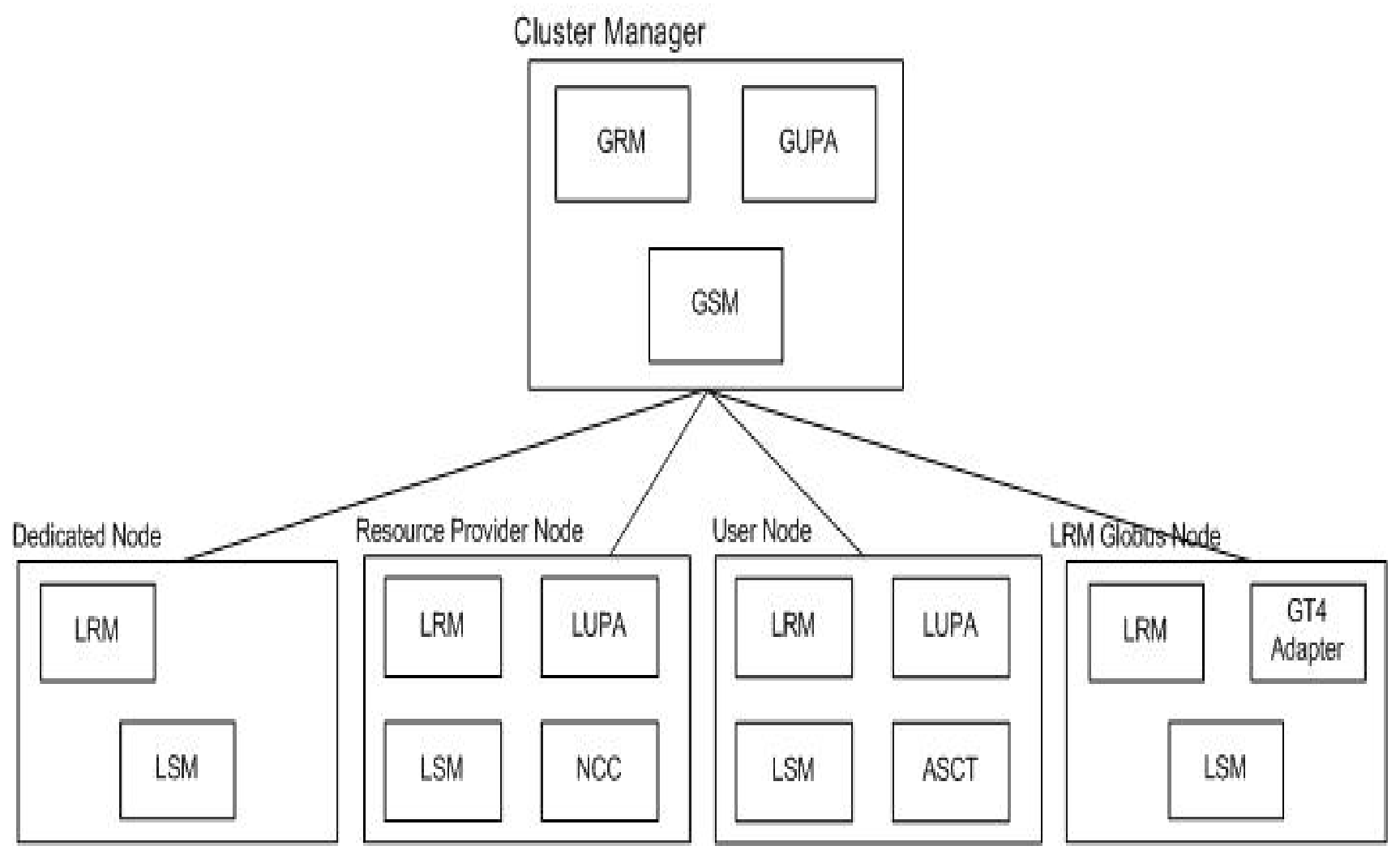

Figure 4 - Globus like InteGrade component

Preliminary Results

The numerical results presented were obtained as follows: a script in Perl was developed on the Globus side and a Java module was developed on the InteGrade side. A performance analysis was performed based on the execution time of the fibonacci(45) function on up to 4 machines. Figures 5 and 6 illustrate the results.

Globus and InteGrade Analysis

The InteGrade performance is a little worse than that of Globus because it needs to register the application in the application repository and each LRM has to search the application bytecode whenever this information is necessary. On the other hand, Globus entirely transfers all the application code to each remote host that will execute the task.

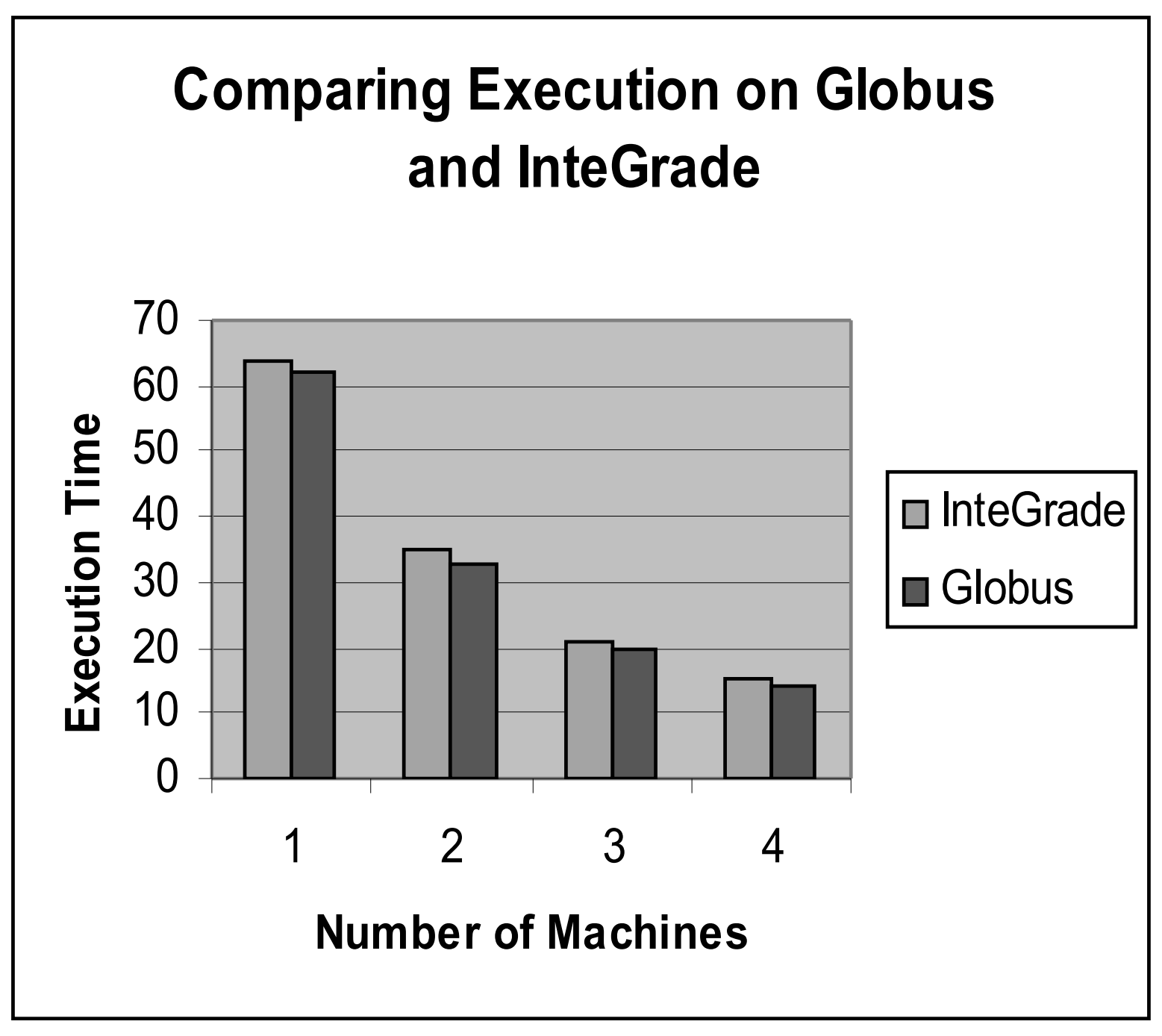

Figure 5: Globus and InteGrade performances

Analysis of the Integrated Environment

Figure 6 shows the execution time of the same application using the integrated system, that is, submitting an application to InteGrade from Globus or to Globus from InteGrade. The execution of the application using the integrated environment does not significantly modify the execution times. It does not matter if the application is executed from Globus or from InteGrade. There is a higher delay compared to the execution in each environment separately, because it is necessary to wait for GRAM, on the Globus side, and for ASCT, on the InteGrade side, to answer to the availability request so that they can execute the tasks. This happens due to the relative simplicity of the application used as example. However, with more complex applications, which consume larger amounts of resources, the influence of this overhead tends to diminish and the relative performance of the integrated system tends to increase.

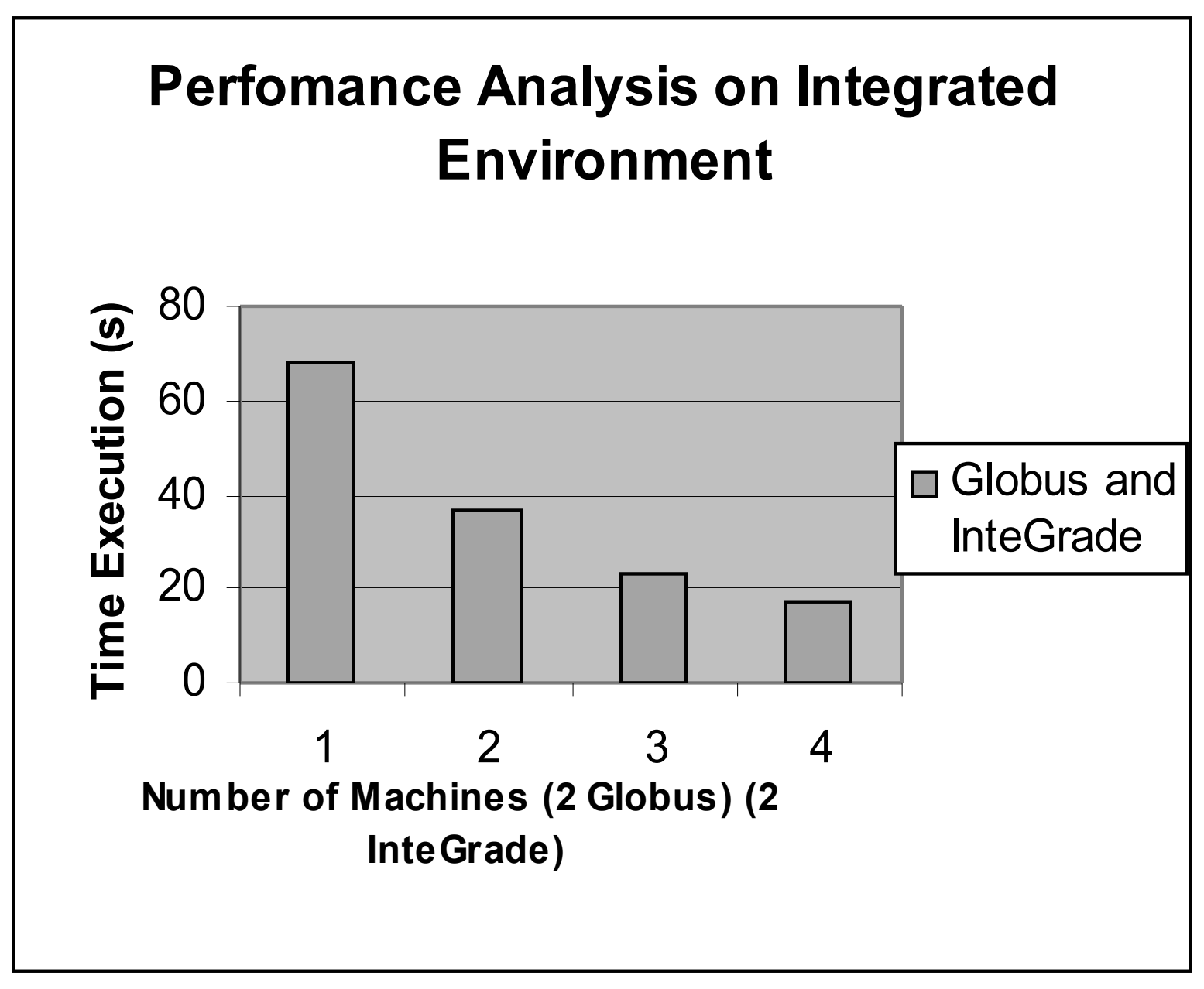

Figure 6 - Integrated environment performance

Related Work

Several research groups have devoted efforts in order to integrate different kinds of grid computing environments. Such efforts address collaboration, data sharing and interaction between distributed resources.

Job submission, monitoring and control using Condor-G and Globus [Mausolf 2005] show how we can use Condor job submission components with Globus to provide a more comprehensive solution. Condor is a batch queuing system and job scheduler for managing compute-intensive jobs, developed at the University of Wisconsin at Madison. Its combination with Globus provides a good interface for job submission, monitoring, and control in a heterogeneous compute environment that may span administrative domains. The Java CoG project [Verma 2001] has done some similar adaptation between different (OGSA and pre-OGSA) versions of Globus. Another work is the GIN consortium (Grid Interoperability Now) carried out by various researchers at the Global Grid Forum (now called the "Open Grid Forum") [GIN 2006].

The D-Grid integration project [DGRID 2005] of the D-Grid consortium is an ongoing effort to build a platform for distributed collaboration on top of Internet-based Services. This platform is built around general middleware and network infrastructure, allowing the provision and demanding of Grid resources (i.e. computing power, data, information, programs, etc.) and its dynamic allocation. The aim is to build the basis for new community-specific applications, enabling new interdisciplinary collaborations to solve scientific problems. On top of this infrastructure new kinds of "virtual organizations" will be created and will benefit from the general availability of services, resources and infrastructures.

Our work complements the above mentioned efforts by defining an architecture for the seamless integration of grid environments that use different programming models, enabling grid heterogeneity at the middleware level.

Final Remarks

The main goal of this work was to develop an interface to integrate heterogeneous grids, using as case study the Globus and InteGrade environments. The integration architecture is defined and implemented as a set of modules, as part of both systems, allowing their integration. Among the main motivations of this work, we can mention the need to create and develop a high performance parallel programming environment that could be executed in a simple, efficient, and transparent way, to comprise a larger set of machines. The results obtained from the implementation of the proposed architecture show the feasibility of the integration between the Globus and InteGrade tools and the several advantages of this integration, such as: the mapping of a new component in InteGrade (LRM-GT4) makes it possible to create a new module for InteGrade, which is able to interact with a Globus-based grid and understand its requirements; the possibility for InteGrade to use the MDS and GSI components through the interaction with Globus, allowing it to use new features to distribute and locate new computational resources; the integration and the interchange of knowledge between researchers of different research centers that eventually use different grid middleware platforms. As future extensions of this work, we aim to explore issues such as load balancing, whose algorithm can be modified in order to make it more efficient, and access to remote files, as well as to perform further tests, using more complex applications. Web Services can also be used to implement future integrated features. Finally, we also aim to extend the architecture so that it can be applied to integrate other grid middleware platforms as well.

Acknowledgements

This work has been partly funded by CNPq (The Brazilian Council for Scientific and Technological Development), grants number 478.620/2004-7 and 550.094/2005-9, by Computer Science Institute of Federal University of Alagoas and by FAPEAL (Research Support Foundation of the State of Alagoas, Brazil).

References

[Globus 2006] The Globus Alliance. Available at: <http://www.globus.org> Accessed: February/2006

[Integrade 2006] InteGrade project. Available at: <http://incubadora.fapesp.br/projects/integrade > Accessed: February/2006

[Foster 2001] Foster, I., Kesselman, C., Tueke, S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of Supercomputing Applications, 15 (3), 200-222, 2001.

[Foster 2004] Foster, I. (ed.), Kesselman C (ed.). The Grid 2: Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers: San Francisco, 2004.

[OGSA 2005] Open Grid System Architecture. Available at: <http://www.globus.org/ogsa> Accessed: August/2005

[OMG 2003] OMG. The Common Object Request Broker: Architecture and Specification, rev. 3.0, Object Management Group, Needham-MA, USA, 2003.

[OGSI 2005] Open Grid System Infrastructure. Available at: <http://www.globus.org/toolkit/draft-ggf-ogsi-gridservice-33_2003-06-27.pdf >.

[Condor 2005] Condor Project. 2005 Available at:<http://www.cs.wisc.edu/condor/>. Accessed: August/2005.

[Verma 2001] S. Verma, J. Gawor, G. v. Laszewski, and M. Parashar, A CORBA Commodity Grid Kit, presented at 2nd International Workshop on Grid Computing in conjunction with Supercomputing 2001 (SC2001), Denver, Colorado, 2001.

[GIN 2006] GIN - Grid Interoperability Now. Available at <http://wiki.nesc.ac.uk/read/gin-jobs>. Accessed: September/2006.

[Ourgrid 2005] Ourgrid Project. 2005. Available at:<http://www.ourgrid.org>. Accessed: August/2005.

[Finger 2003] Finger, M., Goldchleger, A., Kon, F., Goldman, A., Bezerra, G. C. Object-Oriented Grid Middleware Leveraging Idle Computing Power of Desktop Machines. Journal of Concurrency and Computation: Practice and Experience, 2003.

[Mausolf 2005] Mausolf, J. Communicating outside the flock, Part 1: Condor-G with Globus – Job submission, monitoring and control using Condor-G and Globus. IBM Technical Report. Available at <http://www-128.ibm.com/developerworks/grid/library/gr-condorg1/>. Accessed: May/2007.

[DGRID 2005] D-Grid Initiative: Integration Project (DGI). Available at: <http://www.d-grid.de/>. Accessed: May/2007