Grid Virtual Laboratory Architecture

Eduardo Grosclaude, Francisco López Luro, and Mario Leandro Bertogna

Department of Computer Science
Universidad Nacional del Comahue
C.P 8300. Buenos Aires 1400. Neuquén. Argentina
{oso,flopez,mlbertog}@uncoma.edu.ar

Abstract. This work describes an approach to managing networked virtual and physical resources under a Grid-based Virtual Organization, viewed as elementary components of Virtual Remote Laboratories. While keeping overhead at local organizations at a minimum, we seek to obtain secure and dynamic configuration of available resources, then providing interactive access to these resources through web interfaces. These resources can be involved in tasks such as parallel computing, internetworking simulation laboratories, etc. Two test cases, representative of two broad use case classes, are implemented.

Keywords: Grid, Resource Management, High Performance Computing, Virtual Laboratory, Distance Education.

1 Introduction

Availability and integration of geographically spersed technological resources is currently one of the most important challenges. The quest for this kind of solutions calls for greater software and hardware requirements. A greater complexity in administration is also brought in, as resources across distributed environments are heterogeneous and a security posture is to be kept. Physical and virtual resources are logically grouped into Virtual Laboratories. These are defined by UNESCO[1] as electronic workspaces to collaborate and experiment in research and other creational activities, for generating and delivering research results using distributed information technologies. The biggest motivations for Remote Laboratories are their ability to scale up, to globally integrate organizations, to allow for sharing of specific resources, for collection and analysis of geographically distributed data, and for interdisciplinary specialists to cooperate. Among common uses for Virtual Laboratories are remote monitoring of production processes, remotely assessing performance of real and simulated facilities, remote configuration and management, and distance education applications. This work proposes a solution for the generation of Virtual Laboratories, using physical and virtual devices deployed at distinct local organizations and logically grouped by a virtual organization over a Grid environment. This must allow for dynamic and flexible configuration of a workspace by means of open standards and protocols.

L. Bougé et al. (Eds.): Euro-Par 2007 Workshops, LNCS 4854, pp. 164–173, 2008.

We will describe the solution's architecture and components, and two use cases will be analyzed: a parallel computing environment (PARCOMP) and an educational remote virtual laboratory in internetworking (NETLAB).

2 Architecture Overview

From the design viewpoint, the proposed architecture is conceptually divided into three layers or tiers. In the first one, named *access tier*, the clients accessing the system are defined. The second, *management tier*, considers access control and creation of resources. Finally, the third, *resource tier*, deals with the implementation of physical and virtual resources.

A diagram of this architecture (fig. 1) pictures clients accessing a virtual organization through Internet. This particular Virtual Organization is composed by four Grid nodes, distributed into two physical organizations A and B. Both organizations own different types of Grid-available resources, but they both have dedicated clusters allowing the instantiation of virtual machines.

The solution we present here produces a space where independent virtual networks can be created. Administration of these networks runs completely apart from the physical networks that support them. Coordination tasks among local administration people are kept to a minimum.

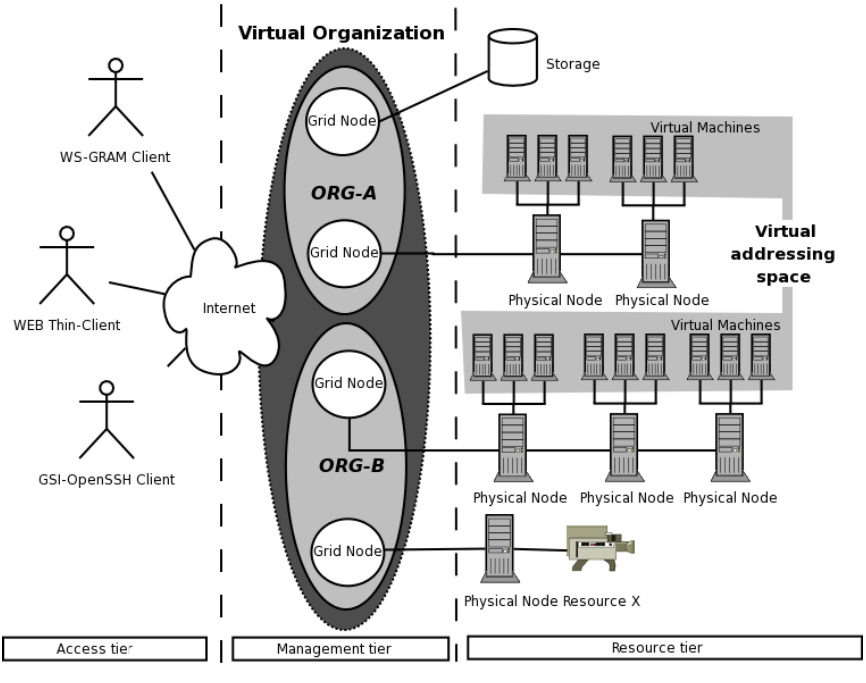

Fig. 1. Architecture

The first type of clients are Grid clients, which give access to resources at one or more Grid nodes from a command line interface. Grid nodes and their resources are managed with a Grid client. Web clients, having no Grid-specific tool or library to execute their tasks, are a second type. They do not know about implementation issues. Final users of the system sit at these clients, and their workspace is a virtual space. Finally, secure clients accesing remote terminals use a Grid security model. This is a special client of the first type. Teachers, or Grid administrators, use this kind of client.

The modules managing the Grid platform are found at the second tier. Each Grid node act as a gateway to the local cluster's private address space. Tasks execution in this space is always done through these nodes. To do work upon physical and virtual machines, command files can be executed in the domains managed by these Grid nodes.

At the third tier, the system's physical and virtual nodes are found. Physical nodes are dedicated to services requested by the Grid node. Each of these physical nodes can instantiate one or more resources such as virtual machines or simulated internetworking devices. Moreover, special resources like physical devices or persistent storage can be accessed from the virtual addressing space.

3 Architecture Components

We schematize in fig. 2 how a client accesses the system services involved, for instance, in the execution of a given request for computing nodes. Here the client calls a scheduling service, parametrically expressing his computing needs in an XML-based description language. The result from this stage specifies which physical nodes will be involved in the service, how many virtual nodes will be spawned at each of them, and which implementation of virtual machines will support each virtual node. These results are then sent by the scheduler as a parameter to the corresponding Web Service at each Grid node. Other parameters included in the scheduler's results are the range of network addresses to be used; and the service access points, if interactive access is in order. Each Grid node will instantiate the machines and return a service access locator for the client to access this network of virtual resources.

3.1 Requirements Specification

The system's input, at the edge next to the user, is a description of the virtual scenario the user intends to instantiate. The feasibility of this virtual scenario will be ensured, and then the proper logical resources will be mapped over a set of physical resources. Ideally, the user will operate an interface for edition of requirements to build the original description. For simplicity, to test our architecture we have devised a small programming interface which suffices to demonstrate the system's usage. This programming interface implements a simple set of classes modeling the desired devices and topology, as well as their mapping to the devices in the physical plant.

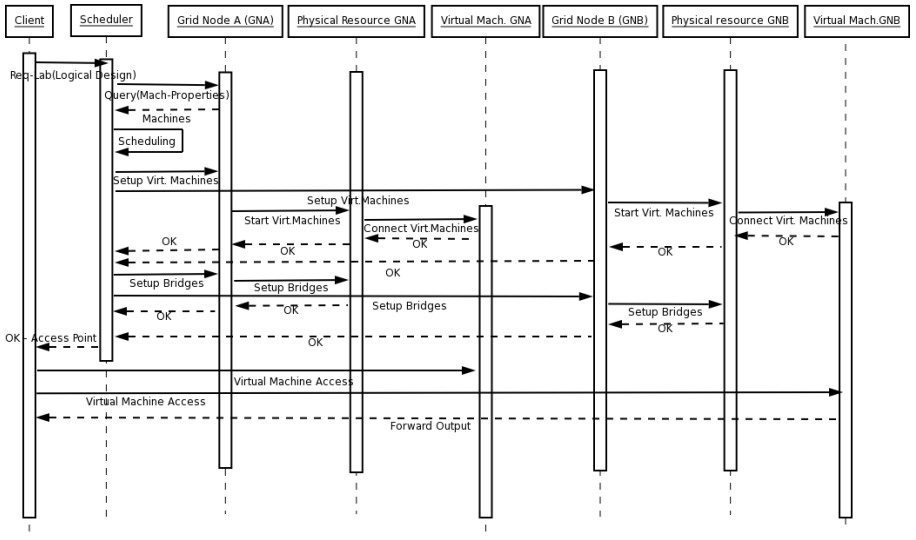

Fig. 2. Sequence Diagram

Using an object based language, the user can describe her scenario, essentially a graph whose nodes are the virtual nodes in the logical network to be instantiated; and whose arcs are the links that build up the desired topology. The scheduler will later resolve the request by mapping the virtual elements in the scenario to certain elements in the physical plant.

The programming interface assisted us during the tests to manually specify suitable scenarios and their mappings. This compensated for the lack of a visual interface and allowed to bypass the scheduling stage at the same time.

3.2 Scheduler

Grid nodes have status information about hosts in the physical plant. The scheduler, by reasoning upon configuration and status data about those physical hosts, will output an allocation plan or mapping between virtual and pyhsical nodes. A given Grid node will offer a variable but limited amount of resources. Under non-availability or high load conditions, the scheduler can instantiate logical resources at several other Grid nodes if needed, then building a virtual network among them. The ability to detect which physical machines in the Virtual Organization have virtualization software is needed to carry on the scheduling phase.

In addition to building the mapping plan, the scheduler must return the service access locators for the virtual elements instantiated and other physical devices or services. For the PARCOMP use case, the (*sparse*) mapping will instantiate no more than a single virtual machine per physical host; all these virtual machines will share the same addressing space; and the output to the user will

be just a single service access locator and a service URI for persistent storage. On the other hand, for the NETLAB use case, the (*dense*) mapping will work under fundamentally different scheduling constraints, i.e. possibly requesting many virtual resources per physical host. These virtual resources will also live in the same addressing space but the scheduler's output to the user will be a set of service access locators and URIs, one for each virtual resource instantiated.

For the NETLAB use case, physical topology can be exploited according to logic topology (i.e. mapping certain virtual links to certain physical links). In the PARCOMP use case, it makes little sense to instantiate more than one virtual machine on the same physical machine, as the application is performance-driven; and the whole instantiated scenario should be located into a same physical cluster if possible.

3.3 Logical to Physical Mapping

The monitoring and discovery system offered by the Grid environment allows users to know which resources are considered a part of the Virtual Organization, and to monitor their status. Every piece of information acquired through aggregation services is maintained in XML and accessed through XPath queries. The same can be said about other query mechanisms through Web Services.

In our work, the information flow hierarchy begins at the cluster monitoring level [2]. The Ganglia package was used to link cluster data to each one of the Grid nodes. To acquire information about active machines in the clusters, the scheduling service sends an XPath query to the Grid node. This query is generated following the requirements presented in the logical scenario and can contain threshold specifications such as free memory or idle CPU status.

3.4 Resource Instantiation

Once a feasible logical-to-physical mapping is obtained, and having previously ensured the availability of resources, virtual machines are created in physical nodes under each Grid node. As shown in fig. 2, the scheduler will send allocation requests to every Grid node using a Web Services interface. The Grid nodes in turn are to redirect these requests to corresponding physical nodes. Scripts were implemented to translate the specifications for virtual machine instantiation into configuration files to be interpreted in each physical node. Extending these scripts transparently allows for the use of new virtualization technologies.

3.5 Network Virtualization

After every logical resource requested in the original scenario is up and running, they have to be interconnected so as to reflect the required topology. However, we seek to hide the details concerning the particular networks and physical hosts existing behind each Grid node. Therefore, we need a private network to relate the virtual machines. We selected VDE_SWITCH [3] and NetCat[4] as tools to deploy virtual "cabling". Custom scripts were created to build up a

solution according to the system's needs. Virtual machines are linked by means of "virtual cables" using TCP tunnels as a transport between physical nodes. A similar approach is used to interconnect LANs on different domains, this time under an SSH connection to ensure privacy. Level 2 frames exchanged by the virtual machines are encapsulated into the tunnel. The elements in the instantiated scenario stay transparently interconnected over a private virtual addressing space which cannot collide with other addressing schemes at the participating Grid nodes.

4 Use Case Implementation

The test cases selected for this work have different goals, and their corresponding scenarios present different topologies and interaction requirements. In the NETLAB test case, links do have specific attributes, such as bandwidth, delay or reliability, and they are crucial to the functioning of the instantiated scenario. Access to the instantiated elements by the user is essentially interactive. In the PARCOMP case, the effective processing power of virtual machines is what matters. Tasks are launched in batch mode, and they are not interactive. As a consequence, both scenarios will challenge the mapping process with different goals and constraints.

Our computing environment is composed by two clusters. One of them is located at premises of the Universidad Nacional del Comahue (UNC) and the other one is hosted at a local IT company (CDF). Both locations are in the same city. Hosts in the UNC cluster are Pentium IV 2.2 MHz machines with 512 MB RAM. The CDF cluster is composed by five Pentium IV 3.06 GHz, HyperThreading machines, with 1GB RAM. The software used for managing the Grid environment is Globus Toolkit 4 [5]. The operating system was Linux, distributions CentOS and Fedora Core 6. The PARCOMP use case was implemented on Xen 3.0.3 [6] virtual machines. For the NETLAB use case, QEMU [7] virtual machines were also used.

4.1 Parallel Application Use Case (parcomp)

The goal for the PARCOMP use case is testing the practical feasibility, and measuring the overhead, of our solution for the execution of parallel applications. We want to benefit from a Grid environment's capabilities, while not having to modify the application. For this use case we selected a parallel application used to model transmission in neural synapses. This application belongs to former work developed by the Complex Systems research group at Universidad Nacional General Sarmiento[8]. The application works under the master-worker paradigm, adequately balancing load among the workers. After each Worker finishes a batch of work, it reports partial results to the Master. The Master then collects the partial results and proceeds with other sequential processing [9].

The application was run using the MPICH 1.2.7p1 parallel library. Three kinds of tests were performed. The first test consisted in the execution of the

application without involving any virtualization. The goal of this first test was to acquire a baseline against which virtualization overhead could be measured. The next test uses virtual machines over physical machines, but no network virtualization. This test allows us to observe application processing overhead. The last execution test uses virtual machines as well as virtual networking, and allows us to know the amount of communications overhead in the complete solution.

As can be seen in Table 1, the Worker's computing overhead for the virtual computing-only test is very low. However, measurements taken at the Master (which account for coordination and data exchange with Workers) show a noticeable increment in computing time for the different virtualization types.

Table 1. Parallel Aplication Benchmark

	Master	Worker
Pure Parallel	512 secs.	50 secs.
Virtual Machines -Physical Network	640 secs.	69 secs.
Virtual Machines - Virtual Network (With Netcat)	768 secs.	73.5 secs.
Virtual Machines -Virtual Network (With SSH)	768 secs.	74 secs.

4.2 Internetworking Laboratory Use Case (netlab)

The NETLAB use case consists in a computer internetworking scenario. Students are expected to practice some network configuration and administration techniques inside this environment. We assume resources are limited at the university labs, hence machines hosted at remote facilities owned by an external entity (such as a partner company) are made available under an agreement and accessed through Internet. To this end, the partner entity stores a repository of system images that can be used to do networking practices. The goal for this use case is to show how the proposed infrastructure can allow the automatic instantiation of virtual laboratory scenarios using a Grid environment and making use of physical resources available at each organization.

As a simple example, the practice assignment implies interconnecting three nodes, located on two different LANs linked by a router. First, the teacher generates a template with the logical design of the assignment. The particular amount of physical resources available at the university labs is immaterial at this stage. Then she invokes the scheduling service of the solution for every student group taking the assignment. The scheduler will instantiate virtual machines on physical hosts across the virtual organization as needed. The virtual scenarios in execution will be the resulting workspaces. The virtual devices are made available through their service access locators, published on web pages dynamically generated. The students, using a Java-enabled browser, access the resources from different locations and collaborate on the assignment.

The experiences showed variable delays, depending on the link capacity and offered load at the moment of performing the assignment work. Performance

measurements were done using the NetPerf tool[10]. As shown in fig. 3, the non-virtualized execution, at 98 Mbps, is close to the theoretical expectation. As connections are virtualized, transmission capacity loss reaches about 50% over an SSH "cable".

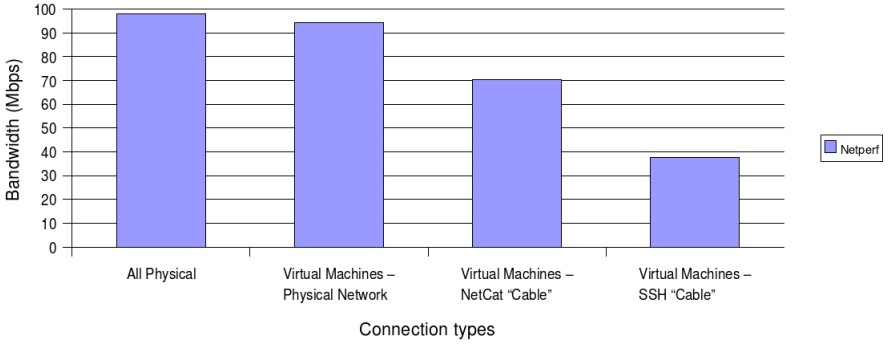

Fig. 3. Virtual Network Benchmark

5 Related Work

Some amount of work has been published about Virtual Laboratories, usually showing high performance environments as a use case, and highlighting some aspect of the solution. The main concern of the Virtual Spaces project [11] is defining and administrating virtual spaces in a Grid environment. Its use case is based on clusters of virtual machines inside a local area network. Cluster on Demand[12] implements the packaging of a cluster scheduler to obtain subsets of a physical cluster through dynamic network address assignment. VioCluster[13] is closely related to this work, although it does not consider Grid for virtual network configuration, nor dynamic discovery of candidate physical machines for virtual machine instantiation. Instead, VioCluster focuses on automatic negotiation of administration domains following established policies, and relates to autonomic concepts. In-VIGO[14], at a very higher abstraction level, allows applications to use virtual environments through Grid services.

Our work can be compared to these efforts, as all of them seek to provide virtual environments through the use of virtual machines. Our approach differs from them in that Grid technology is used to build the communication infrastructure for resources, allowing the dynamic creation of common spaces across different domains.

6 Conclusion

We described a feasible solution for the coordinated usage of geographically spersed computing resources, under a virtual organization scheme provided by

the Grid infrastructure. The solution makes use of virtual machines for flexibility and transparency.

Our work delivers an approach to issues related to configuration, access and resource management. Regarding configuration, a simple object-based language has been specified for the design and validation of the logical requirements. As for access, a model for service access locators, achieving virtual terminal redirection from the resources across Internet, onto the user, has been developed and tested. As for management, Grid middleware functionality has been used to allow remote execution and data transfer across administrative domains with no work required from local administrators.

Use case implementation allowed us to gather some experience about the proposed solution's behaviour under different sets of requirements. The first one, the use of a parallel application, where network configuration is trivial, with a single type of virtual machine but with strong performance requirements. The second one, a remote laboratory where the main requirement is flexibility in the configuration of several kinds of scenarios, and there is a demand for a high number of virtual machines, but no strong performance requirements.

The implementation of these two use cases motivated a detailed analysis. The existence and separate configuration of the private addressing space provided addressing transparency and allowed the enforcement of a security policy when working on lab assignments (NETLAB). Usage of virtual machines, when implementing a parallel computing application (PARCOMP), eased working with divergingly configured clusters (such as when having different versions of parallel libraries) with little administration overhead. In both cases, introducing a Grid environment into the problem of cluster hosts configuration allows for the combination of a greater number of resources on demand, therefore enhancing usage patterns.

In the case where performance requirements are not specific (NETLAB), and where primary importance is given to the creation of a virtual work environment, separated from the encompassing physical environment, this is an acceptable solution which allows users to make an efficient usage of available resources, or introduce otherwise unavailable elements. When the application bears performance constraints (PARCOMP), usage analysis must lead to further studying the infrastructure behaviour for the particular application (namely, computing power demand, amount of coordination messaging, bulk data transfer volume, network virtualization overhead must all be taken into account).

Our future work includes studying the optimization of the virtual machines' networking. Enhancing performance at the point of access to the network is of prime importance for the parallel computing class of use cases. For Xen virtual machines, this is currently dependent on Linux "bridge" devices. As for configuration, we plan to extend the specification language, introducing new interfaces. As for administration, we plan to define a framework for querying virtual scenarios and for dynamic creation of virtual machines on a package- or service-provided basis, so as to be able to easily manage repositories to enhance the solution's availability and flexibility.

References

1. James, P., Vary, E.: Report of the Expert Meeting on Virtual Laboratories. Technical Report CII-00/WS/01, International Institute of Theoretical and Applied Physics (IITAP), UNESCO (2000)
2. Sacerdoti, F.D., Katz, M.J., Massie, M.L., Culler, D.E.: Wide area cluster monitoring with ganglia, pp. 289–298 (2003)
3. Davoli, R.: Vde: Virtual distributed ethernet. In: TRIDENTCOM 2005: Proceedings of the First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities (TRIDENTCOM 2005), Washington, DC, USA, pp. 213–220. IEEE Computer Society, Los Alamitos (2005)
4. NetCat, `http://netcat.sourceforge.net/`
5. Foster, I.T.: Globus toolkit version 4: Software for service-oriented systems. In: NPC, pp. 2–13 (2005)
6. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the art of virtualization. In: SOSP 2003: Proceedings of the nineteenth ACM symposium on Operating systems principles, pp. 164–177. ACM Press, New York (2003)
7. QEMU, a fast and portable dynamic translator (2005)
8. Carusela, M., Perazzo, R., Romanelli, L.: Information transmission and storage sustained by noise. Physica D 168–169, 177–183 (2002)
9. Argollo, E., Gaudiani, A., Rexachs, D., Luque, E.: Tuning application in a multicluster environment. In: Euro-Par, pp. 78–88 (2006)
10. NetPerf., `http://www.netperf.org/netperf/`
11. Foster, I.T., Freeman, T., Keahey, K., Scheftner, D., Sotomayor, B., Zhang, X.: Virtual clusters for grid communities. In: CCGRID, pp. 513–520. IEEE Computer Society, Los Alamitos (2006)
12. Chase, J.S., Irwin, D.E., Grit, L.E., Moore, J.D., Sprenkle, S.E.: Dynamic virtual clusters in a grid site manager. In: HPDC 2003: Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing (HPDC 2003), p. 90. IEEE Computer Society, Los Alamitos (2003)
13. Ruth, P., Mcgachey, P., Xu, D.: Viocluster: Virtualization for dynamic computational domains. In: Proceedings of the IEEE International Conference on Cluster Computing (Cluster 2005) (2005)
14. Adabala, S., Chadha, V., Chawla, P., Figueiredo, R., Fortes, J., Krsul, I., Matsunaga, A., Tsugawa, M., Zhang, J., Zhao, M., Zhu, L., Zhu, X.: From virtualized resources to virtual computing grids: The in-vigo system. Future Gener. Comput. Syst. 21(6), 896–909 (2005)