Enabling Execution of Service Workflows in Grid/Cloud Hybrid Systems

Luiz F. Bittencourt, Carlos R. Senna, and Edmundo R. M. Madeira
Institute of Computing
University of Campinas - UNICAMP
P.O. Box 6196, Campinas, São Paulo, Brazil
{bit, crsenna, edmundo}@ic.unicamp.br

***Abstract*—Cloud computing systems provide on demand access to computational resources for dedicated use. Grid computing allows users to share heterogeneous resources from multiple administrative domains applied to common tasks. In this paper we discuss the characteristics and requirements of a hybrid system composed of both grid and cloud technologies. We propose an infrastructure which is able to manage the execution of service workflows in such system. We show how the infrastructure can be expanded by acquiring computational resources on demand from the cloud during the workflow execution, and how it manages these resources and the workflow execution without user interference.**

I. Introduction

The cloud computing brings the supercomputing to the users, making them to transparently achieve virtually unbounded processing and storage accessible from their laptops or personal computers. Through the virtualization, the cloud offers computational resources with different hardware configurations, as the amount of memory or number of cores, accessible through the Internet. The cloud computing provides on demand hardware, allowing enterprises to execute their applications without the need of new investments, converting fixed costs in expenses based on current needs.

In the cloud computing paradigm details are abstracted from the users. They do not need knowledge of, expertise in, or control over the technology infrastructure about the cloud they are using. It typically involves the provision of dynamically scalable and often virtualized resources as a service over the Internet [1]. The cloud computing characteristics are on demand self-service, ubiquitous network access, independent resource location (reliability), rapid elasticity (scalability), and pay per use. The cloud computing allow the use of Service Oriented Computing (SOC) standards, permitting users to establish links between services, organizing them as workflows instead of building traditional applications using programming languages.

The on demand computing offered by the cloud allows users to keep using their particular systems (computers, clusters, and grids), aggregating the cloud resources as they need. However, this technology union results in a hybrid computing system, with new demands, notably in resource management. Besides that, even though it uses the SOC paradigm, the cloud does not offer support to dynamic service workflow composition and coordination.

In this paper we discuss the characteristics of a hybrid system, composed of the union of a grid with a cloud, and we propose an infrastructure able to manage the execution of service workflows in such system.

This paper is organized as follows. Some basic concepts and related works are presented in Section II, while Section III shows the infrastructure to execute service workflows in the hybrid system. Section IV presents the system architecture, and application scenarios are discussed in Section V. Conclusion and future works are shown in Section VI.

II. Concepts and Related Works

Our infrastructure is directed to systems that use the service oriented computing paradigm. In our work we combine a service oriented grid and the Nimbus cloud [2], an option based on the Amazon's Elastic Compute Cloud (EC2) [3]. In this context, some concepts and standards are important, and it is convenient to describe them.

Grids are environments where shared heterogeneous computing resources are connected through a network, local or remote [4]. Grids allow institutions and people to share resources and objectives through security rules and use policies, comprising the so called Virtual Organizations (VOs) [4]. The Open Grid Services Architecture (OGSA) standard [4] proposes that the interoperability among grid heterogeneous resources to be made through Internet protocols, allowing grids to use standards and paradigms from the service oriented computing (SOC) [5]. In our work we used the Globus Toolkit version 4 (GT4), an OGSA implementation from the Globus Alliance [6].

The grid and virtual organization dynamics intensify the need of on demand provisioning, where organizational requirements must guide the system configuration. It is an environment duty to dynamically provide the services related to each application when they are needed. It is not recommendable to make all services available in all resources in the grid, since this can overload resources and use processing power, memory, and bandwidth without need. To allow on demand provisioning, it is necessary to have support to dynamic instantiation of services, i. e., to send the service to the resource, publish it so it can be handled by a container, and activate the container so it can start replying to service requisitions. Our infrastructure aggregates functionalities for on demand service provisioning during workflows execution, since the GT4 does not offer such functionality.

The cloud computing paradigm abstracts details from users who no longer need knowledge about the technology infrastructure that supports the cloud. It typically involves the provision of dynamically scalable and often virtualized resources as a service over the Internet [1]. Cloud computing delivers three defined models: software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS). In SaaS the consumer uses an application but does not control the host environment. Google Apps [7] and Salesforce.com [8] are examples of this model. In PaaS the consumers use a hosting environment for their applications. The Google App Engine [9] and Amazon Web Services [10] are PaaS examples. In this model the platform is typically an application framework. In IaaS the consumer uses computing resources such as processing power and storage. In this model the consumer can control the environment including deployment of applications. Amazon Elastic Compute Cloud [3], Globus Nimbus [2], and Eucalyptus [11] are good examples of this model.

Nowadays, the most popular model is the IaaS. In a simplified manner we can understand this cloud model as a set of virtual servers accessible through the Internet. These servers can be managed, monitored, and maintained dynamically and remotely. It is easy to see that the virtualization concept is fundamental in the IaaS model. Virtualization [12] is the process of presenting a logical grouping or subset of computing resources so that they can be accessed in abstract ways with benefits over the original configuration. The virtualization software abstracts the hardware by creating an interface to virtual machines (VMs), which represents virtualized resources such as CPUs, physical memory, network connections, and peripherals. Each virtual machine alone is an isolated execution environment independent from the others. With this, each VM can have its own operating system, applications, and network services. This isolation allows users to have control over the resource without interference in other participants in the cloud.

Clouds and grids are distinct. Clouds provide full private cluster, where individual users can access resources from the pool, and its resources are "opaque", being accessible through the user interface without knowledge about hardware details. Grids permit individual users to select resources and get most, if not all, the resources in a single request. Its middleware approach takes federation as a first principle, and it exposes resources without any preparation or isolation. These differences claim different architectures for each one, and require personalized solutions. Our infrastructure supplies service support offering automatic service deployment in the resources provided dynamically, and controlling the service workflow execution through a workflow manager, interacting with the cloud and the grid in a transparent manner without the user interference.

Some works propose solutions for the execution of workflows in grids or clouds, but only a few considers a hybrid environment. In [13] the authors explore the use of cloud computing for scientific workflows. The approach is to evaluate the tradeoffs between running tasks in a local environment, if such is available, and running in a virtual environment via remote, wide-area network resource access. The work in [14] describes a scalable and lightweight computational workflow system for clouds which can run workflow jobs composed of multiple Hadoop MapReduce or legacy programs. But both works do not offer support for services.

In [15] the authors show issues that limit the use of clouds for highly distributed applications in a hybrid system. However, it has lack of interoperability between different cloud platforms, and it does not offer support to service workflows. The authors propose a hybrid system formed by the DIET Grid and Eucalyptus in [16]. It shows possible ways of connecting these two architectures as well as requirements to achieve this, but it does not support services or service workflows. In [17], the authors show a solution that automatically schedules workflow steps to underutilized hosts and provides new hosts using cloud computing infrastructures. This interesting work extends a BPEL implementation to dynamically schedule service calls of a BPEL process based on the target hosts load. To handle peak loads, it integrates a provisioning component that dynamically launches virtual machines in Amazons EC2 infrastructure and deploys the required middleware components (web/Grid service stack) on-the-fly. However, it does not support hybrid systems.

The work proposed in this paper aggregates the support for service workflow execution in both grids and clouds, offering support to on demand dynamic instantiation and service publication. Its functionalities allows the user to execute abstract workflows, without indicating the resources where each part of the workflow will execute. Besides that, the hybrid systems management allows the use of different clouds with several architectures.

III. The Hybrid System Infrastructure

The on demand computing requires services scalability, flexibility, and availability. To supply these requirements it is important to the infrastructure to offer reconfiguration with the possibility of the deployment of new resources or update of the existing ones without stopping processes in execution. In a hybrid system composed of a grid with the possibility of accessing a cloud computing infrastructure, the workflow management must supply the requirements in some levels. First, it must provide facilities to the user to make submissions without the need of choosing or indicating their localization of the computational resources to be used. Inside the grid boundary, the workflow manager must find the best resources available and, when necessary, it must make the dynamic deployment of services in these resources. On the other hand, inside the cloud boundary, the infrastructure must be able to interact with the cloud interfaces to obtain computational resources. After that, it must be able to prepare these resources according to workflow necessities, making the dynamic deployment of services in the resources inside the cloud. This deployment can be made when local resources are not enough to the workflow necessities. This configuration increases the computational power of the grid without new infrastructure investment, using the on demand computing advantages provided by the cloud.

In this paper we show an infrastructure for the execution of service workflows in hybrid systems composed of grid and clouds. The infrastructure provides dynamic instantiation of services when necessary, and it is formed by a set of services which offer the following functionalities:

- Simple workflow description language: users describe workflows through the Grid Process Orchestration Language (GPOL). The GPOL allows users to build abstract workflows, where the computational resources do not need to be indicated;
- Dynamic service instantiation: during the workflow execution, the infrastructure searches, in the grid and in the cloud, the best computational resources available to execute each service;
- Automatic reference coordination (endpoint reference service): when offering dynamic instantiation, some activities must be transparent to the users. For example, consider a service that when executed generates a file that is used by other services in the workflow. Because the services localization is made on demand, the infrastructure resolves the references between services in execution time, without user interference;
- Dynamic service deployment: when the best resource option is identified, the infrastructure can deploy the new service if necessary. The dynamic deployment is executed regardless the resource is in the grid or in the cloud; and
- Robust workflow execution: if a service fails during the execution, the infrastructure can search for an alternative resource, automatically redirecting the execution and making the necessary adjustments in the services references. If there is no resource with the service available, the infrastructure tries to publish the service in a resource.

The set of services which compose our infrastructure is introduced in the next section, covering the Workflow Manager, the Deployment and Undeployment Service (DAUS), the Scheduling Service (SS), the Resource Monitor and Repositories, and the Dynamic Deployment Virtual Resource (DDVM).

IV. The Infrastructure Architecture

The proposed infrastructure architecture, formed by a set of services, is shown in Figure 1. Its main components are a workflow manager, a scheduler, a service for publication of services, a service for dynamic publication of virtual resources in the cloud, and repositories used by these components.

In a typical execution in the grid, the user submits a workflow to the Grid Process Orchestration (GPO), the workflow manager in the system. The GPO analyses the workflow, consulting the scheduling service (SS), which identifies the best resource available for each service involved. If a new service needs to be published, the GPO makes the new service deployment through the Deployment and Undeployment Service (DAUS), which publishes the new service and returns its localization (endpoint reference) to the GPO. Using the endpoint reference, the GPO creates the instances which are necessary during the workflow execution. When the execution is terminated, the GPO can automatically eliminate the instances created and added to the DAUS, or it can leave the services for further use when necessary. If the workflow execution needs more resources, the GPO starts the Dynamic Deployment Virtual Resource (DDVM), which searches for resources in the cloud. The DDVM requests resources to the cloud informing its localization to the GPO. Through the DDVM, the GPO uses the cloud resources in a similar manner to the use of local resources in the grid. It publishes and instantiates the services dynamically, and it monitors the resources through the infrastructure services.

A. The Workflow Manager

To manage the service composition in our infrastructure we used the GPO [18], a middleware to support interoperability of distributed applications which require service composition in the computational grid. The GPO allows the creation and management of application flows, tasks, and services in grids. Its architecture is based on concepts presented in the OGSA (GT4 implementation), and it extends the GJD (WS-GRAM Job Description Language) [19] with WS-BPEL (Web Service Business Process Execution Language) [20] characteristics, an OASIS (Organization for the Advancement of Structured Information Standards) specification, in the description of the flows to be executed. The workflows are built through the GPOL [18]. The GPOL is an XML-based language that brings together the GJD characteristics and service orchestration concepts proposed by the WS-BPEL. Some specific directives were added, which are necessary to the grid environment, such as: state maintenance, potentially transient services, notification, data oriented services, and group oriented services. The GPOL includes variables, lifecycle, fabric/instance control, flow control, and fault tolerance. Additionally, it allows the user to start the execution of tasks, services and workflows sequentially or in parallel.

B. The Deployment and Undeployment Service (DAUS)

In the workflow execution, the GPO consults the scheduler about the best resource to execute a service. The scheduler can respond indicating a computational resource where the service is not available. In this case, the GPO uses the DAUS that, with information about the resource identification returned by the scheduler, executes the following actions to dynamically publish the necessary service:

- Creates a DAUS instance in the grid or cloud resource;
- Consults the remote instance about the necessity of sending or not the service source code to the remote resource. This verification guarantees that the source code is transmitted only once during the workflow execution. If it is not in the resource, the DAUS obtains the source code from the service repository;
- Requests a port number not used yet by the containers in the remote resource;

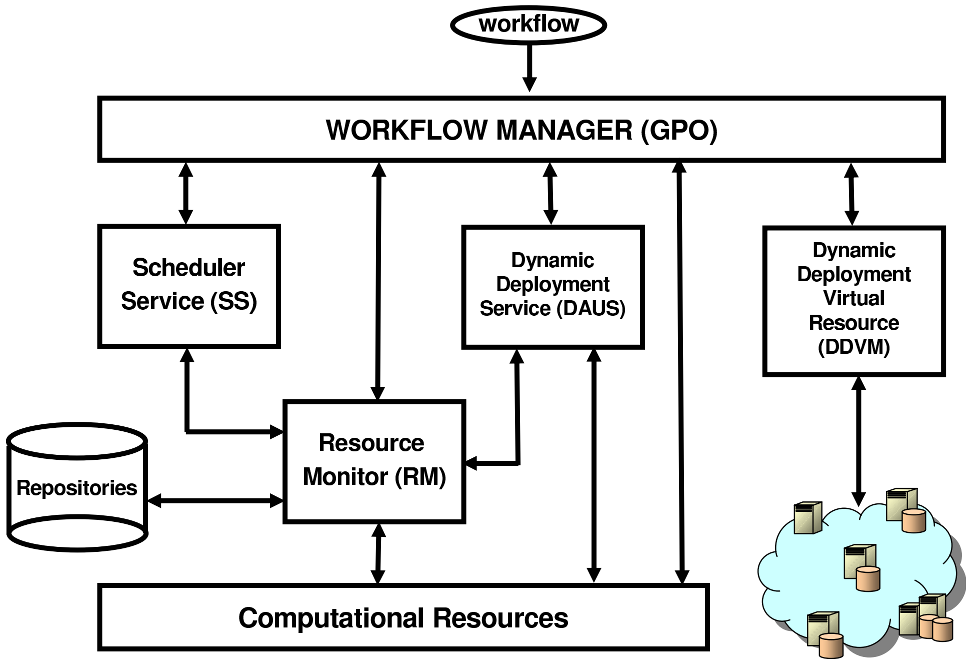

Fig. 1. The proposed architecture.

- Consults the remote instance about the necessity of publishing the new service. The remote instance publishes the service when necessary.
- To not interrupt the containers already executing and to allow the immediate use of the service in the remote resource, the infrastructure starts a new container in the available port; and
- With all actions executed and the new container available, DAUS sends to GPO the localization in the remote resource (URI): Protocol://IPHost:Port/Service_Name.

GPO uses the localization of the published service to create all the instances of the service necessary to the workflow execution. When it is finished, GPO can eliminate the created instances and call DAUS again to eliminate the created container and remove the published services from the remote resource. There is also the option of leaving the dynamically published services for further use.

C. The Scheduling Service (SS)

The scheduling service (SS) shown in Figure 1 has the function of distributing the workflow services to be executed in the grid resources. To do this, the scheduler may use simple algorithms, such as distributing the services according to the number of cores in a given resource, or it can use more complex algorithms ([21], [22]), which make the scheduling decisions based on information provided by monitoring services. In a hybrid grid/cloud system, a more advanced scheduling algorithm for service workflows must consider some peculiarities that arise, such as:

- The time to obtain a resource from the cloud.
- The different bandwidth to transfer services code and data to VMs in the cloud and to resources in the grid.
- Performance degradation related to the overhead inserted by the execution of VMs.
- Overhead related to the dynamic publication of services in the cloud and in the grid, including the time to send necessary infrastructure services, the service source code, time to start a new container, and time to instantiate a new service.
- Monetary costs related to the use of cloud resources instead of grid resources.

This new information can be incorporated to the objective function to be optimized by the scheduler to try to, for instance, minimize the execution time of the workflow (makespan). The trade-off between obtaining a new resource from the cloud and using a currently existing one must be deeply studied. This depends on the performance history of resources returned by a given cloud, on the number of resources needed, on the number of times a given service will be executed, on the load peak times in both the grid and the cloud, and so on.

D. The Resource Monitor and Repositories

The resource monitor (RM) is the service which gathers information about computational resources in the hybrid system, grid or cloud. It operates in a distributed manner, maintaining one instance in each computational resource. Such instances are used on demand by the other services when information about resources are needed. The workflow manager uses the RM to have knowledge about which resources are in the grid at a given time, and the scheduler uses it to obtain information about the current state of the resources. Based on information stored in the RM, the scheduler can simply schedule the unscheduled services aiming at the minimization

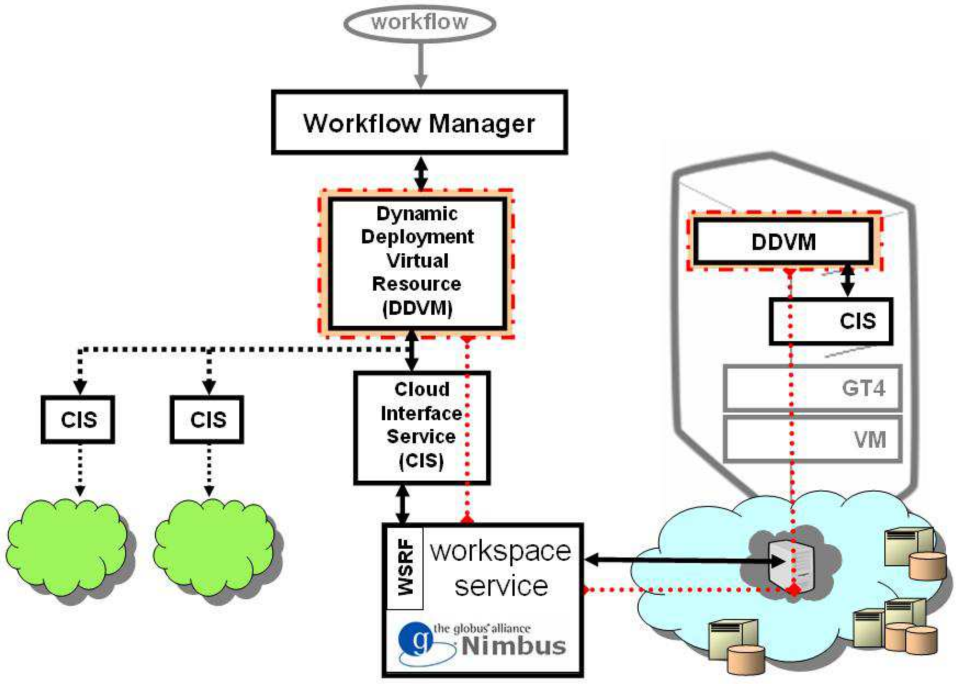
workflow
Workflow Manager
Dynamic Deployment Virtual Resource (DDVM)
CIS
CIS
Cloud Interface Service (CIS)
WSRF
workspace service
the globus alliance Nimbus
DDVM
CIS
GT4
VM

Fig. 2. DDVM and Virtual Computational Resources From Clouds.

of the execution time or it can reschedule some services to achieve a better global performance.

Despite the RM distributed operation, harvesting the current resources state when necessary, some information must be stored in repositories. The resources repository has information such as the characteristics of each computational resource, performance history and load. This information can also be used by the scheduler in its decision process. Besides that, the services repository has information about services available in the grid, and it stores the necessary files to the dynamic publication made by DAUS.

The RM has a strategic function of providing information that allows the resource manager to identify bottlenecks and concurrency situations. The management analyses in a combined manner the VMs load, local or in the cloud, and the load of physical machines where such VMs are hosted. With this information the scheduler can avoid potential overloading, associating VMs with their hosts. For example, without this, the scheduler may have misleading information about the number of free cores, scheduling more processes than the number of available cores in a resource. Another example of bottleneck is the scheduling of processes that make intensive I/O in the same physical resource because of lack of knowledge about where each VM is running.

E. Dynamic Deployment Virtual Resource (DDVM)

The DDVM, shown in Figure 2, is used by the infrastructure when resources from the cloud are needed. It is composed of two groups of services. The first group is the DDVM itself, which communicates with the infrastructure, taking care of the functionalities. The second group, called Cloud Interface Service (CIS), makes the interface with the cloud. This layout gives flexibility and scalability to the infrastructure, allowing the access to different clouds in an independent and simultaneous manner, as illustrated on the left-hand part of Figure 2. For each resource, one instance of the couple DDVM/CIS is responsible for the binding between the workflow manager and the cloud resource. To use the cloud resources, the GPO communicates with the DDVM, which communicates with the cloud through CIS, and requests a resource. The cloud returns the localization of the VM obtained and the DDVM can start using it by initiating the Globus Toolkit and creating a DDVM instance in the cloud resource.

The logic link between components is represented by the dotted line in Figure 2. This dotted line starts on the DDVM instance that is in the grid resource along with the GPO, then it goes through CIS, and it passes through the cloud interface, finally reaching the VM made available by the cloud. In this example we used a Nimbus cloud [2], but other clouds may be attached to the infrastructure. Through the two DDVM instances, the grid one and the cloud one, GPO runs the services that compose the workflow in the same way it does with resources from the grid.

Figure 3 shows the complete infrastructure installed in the hybrid system composed of the grid and the cloud. On the right-hand side there are the components which are installed in the grid resources, where is the workflow manager. On the left-hand side the same components are shown, but as part of resources in the cloud. The binding between these two infrastructures is made by the DDVM/ICS couple. There is no restriction regarding the use of services allocated in the cloud's VM. For example, if necessary, the GPO can use the

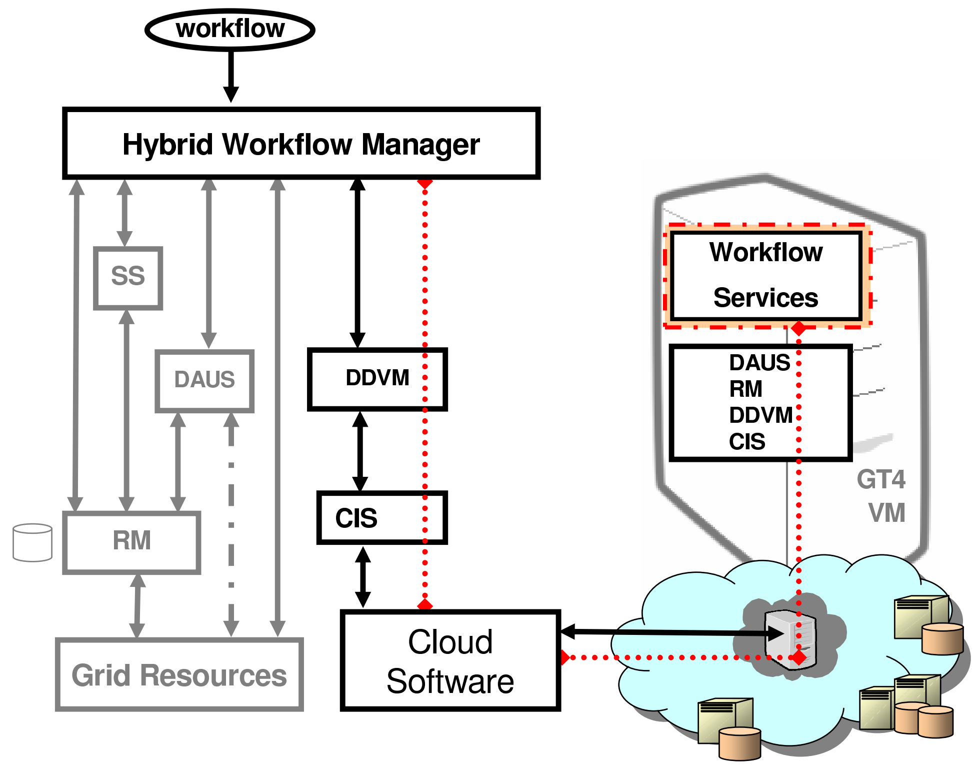

Fig. 3. Hybrid Infrastructure for Service Workflow Execution.

DAUS and dynamically deploy a service which is not yet installed in the cloud's VM. In the same way it can use the RM to obtain information about the cloud resources load or configurations. In Figure 3, the dotted line represents workflow services instantiated in the VM, and which are being used during the workflow execution.

V. Application Scenarios

The proposed infrastructure can be useful in many scenarios that appears in grid application executions nowadays. If we consider the minimization of the makespan as the objective to be achieved, the infrastructure has always the option of requesting cloud resources if the current resources available cannot give a satisfactory makespan. For instance, this can occur if the grid is overloaded with many submissions in a load peak. In this scenario, a new workflow may have its speedup heavily prejudiced because it may need to wait for many other workflows to finish their execution.

Another scenario where we envision that our infrastructure can be applied is when a workflow has more parallel tasks than the number of resources available. An example of application that is represented by a workflow and can have different sizes is Montage [23]. Montage is an image application that makes mosaics from the sky for astronomy research. Its workflow size depends on the square degree size of the sky to be generated. For example, for a 1 square degree of the sky, a workflow with 232 jobs is executed. For a 10 square degrees of the sky a $20,652$ jobs workflow is executed, dealing with an amount of data near to 100 GB. The full sky is around $400,000$ square degrees [24]. For such an application, an elastic infrastructure is desired, where the number of resources can be adapted according to the size of the application to be run. In our hybrid system, the infrastructure can avoid the cloud use when the grid resources are sufficient to the execution of the workflow. On the other hand, when the workflow is too large, the infrastructure can gather resources from clouds to afford its execution.

Furthermore, our infrastructure can be applied in cases where deadlines for the completion of the workflow execution exist. If the scheduler finds that the grid itself is not able to provide resources in the quantity and quality needed to execute a workflow within a given deadline, it may ask the cloud for resources to compose the infrastructure and therefore be capable of finishing the workflow before the deadline is reached.

VI. Conclusion

The cloud computing provides computational resources on demand for dedicated use. On the other hand, the computational grid proposes interoperability among heterogeneous resources through Internet protocols. In this paper we discuss the characteristics and requirements of a hybrid system formed by these two technologies, and we propose an infrastructure to the management of service workflows in this system. Our motivation comes from the fact that both technologies do not offer adequate support to the execution of service workflows, giving to the user the responsibility of preparing the environment for such execution. We propose an infrastructure that covers this aspect, offering support to automatically install services in the resources dynamically provided by the grid or by the cloud, while providing the service workflow execution control. Our workflow management system interacts with the cloud and with the grid in a transparent manner, without the user interference, and its functionalities allow the execution of abstract workflows without indicating which resource must be used. Additionally, the proposed hybrid system management gives flexibility and scalability to our infrastructure, permitting the use of several clouds with different architectures in an independent and simultaneous way.

To improve the efficiency of the infrastructure, future works are needed. Research in scheduling algorithms and strategies to deal with trade-offs between using grid resources and cloud resources are mandatory. Another point to study is how to identify clouds load peaks and choose the best cloud to be used at a given time of the day.

Acknowledgment

The authors would like to thank CAPES, FAPESP (05/59706-3), and CNPq (472810/2006-5 and 142574/2007-4) for the financial support.

References

[1] "The NIST definition of cloud computing 15," National Institute of Standards and Technology (NIST), Tech. Rep., July 2009. [Online]. Available: http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc

[2] "Nimbus toolkit, globus aliance," 2009. [Online]. Available: http://workspace.globus.org/

[3] "Amazon elastic compute cloud (amazon EC2)," 2009. [Online]. Available: http://aws.amazon.com/ec2/

[4] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *International Journal of Supercomputer Applications*, vol. 15(3), pp. 200–222, 2001.

[5] F. Curbera, R. Khalaf, N. Mukhi, S. Tai, and S. Weerawarana, "The next step in web services," *Communications of ACM*, vol. 46, no. 10, pp. 29–34, 2003.

[6] "Globus toolkit version 4, globus aliance," 2008. [Online]. Available: http://www.globus.org/toolkit/

[7] "Google app," 2009. [Online]. Available: http://www.google.com/apps/intl/en-GB/business/index.html

[8] "Salesforce," 2009. [Online]. Available: http://www.salesforce.com/

[9] "Google app engine," 2009. [Online]. Available: http://code.google.com/intl/en/appengine/

[10] "Amazon web services (aws)," 2009. [Online]. Available: http://aws.amazon.com/

[11] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The Eucalyptus open-source cloud-computing system," in *2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID)*, vol. 0. Washington, DC, USA: IEEE, May 2009, pp. 124–131. [Online]. Available: http://dx.doi.org/10.1109/CCGRID.2009.93

[12] J. Smith and R. Nair, *Virtual machines: versatile platforms for systems and processes*. The Morgan Kaufmann, 2003.

[13] C. Hoffa, G. Mehta, T. Freeman, E. Deelman, K. Keahey, B. Berriman, and J. Good, "On the use of cloud computing for scientific workflows," in *ESCIENCE '08: Proceedings of the 2008 Fourth IEEE International Conference on eScience*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 640–645.

[14] C. Zhang and H. D. Sterck, "Cloudwf: A computational workflow system for clouds based on hadoop," in *CloudCom*, ser. Lecture Notes in Computer Science, M. G. Jaatun, G. Zhao, and C. Rong, Eds., vol. 5931. Springer, 2009, pp. 393–404.

[15] S. Jha, A. Merzky, and G. Fox, "Using clouds to provide grids with higher levels of abstraction and explicit support for usage modes," *Concurr. Comput. : Pract. Exper.*, vol. 21, no. 8, pp. 1087–1108, 2009.

[16] E. Caron, F. Desprez, D. Loureiro, and A. Muresan, "Cloud computing resource management through a grid middleware: A case study with diet and eucalyptus," *Cloud Computing, IEEE International Conference on*, vol. 0, pp. 151–154, 2009.

[17] T. Dornemann, E. Juhnke, and B. Freisleben, "On-demand resource provisioning for bpel workflows using amazon's elastic compute cloud," in *CCGRID '09: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 140–147.

[18] C. Senna, L. Bittencourt, and E. Madeira, "Execution of service workflows in grid environment," in *TridentCom 2009: Proceedings of the 5th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops*. IEEE, 2009, pp. 1–10.

[19] I. Foster, "Globus toolkit version 4: Software for service-oriented systems," in *IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779*, Beijing, China, 2005, pp. 2–13.

[20] "Web services business process execution language version 2.0," OASIS Web Services Business Process Execution Language (WSBPEL) TC, Tech. Rep., April 2007. [Online]. Available: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[21] L. F. Bittencourt and E. R. M. Madeira, "A performance oriented adaptive scheduler for dependent tasks on grids," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 9, pp. 1029–1049, 2008.

[22] L. F. Bittencourt, C. R. Senna, and E. R. Madeira, "Bicriteria service scheduling with dynamic instantiation for workflow execution on grids," in *GPC '09: Proceedings of the 4th International Conference on Advances in Grid and Pervasive Computing*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 177–188.

[23] E. Deelman, G. Singh, M.-H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G. B. Berriman, J. Good, A. Laity, J. C. Jacob, and D. S. Katz, "Pegasus: A framework for mapping complex scientific workflows onto distributed systems," *Scientific Programming Journal*, vol. 13, no. 3, pp. 219–237, 2005.

[24] E. Deelman, "Clouds: An opportunity for scientific applications? (keynote in the 2008 Cracow Grid Workshops)," 2008.