GaaS: Customized Grids in the Clouds

G.B. Barone[•], R.Bifulco[•], V. Boccia[♣], D. Bottalico[•♣], L. Carracciuolo[♦♣], R.Canonico[•]

[•]*Università degli Studi di Napoli Federico II*

[♣]*Italian National Institute of Nuclear Physics*

[♦]*Italian National Research Council*

The Grid model is basically static and users cannot define new Grid sites, add computational resources to existing ones and modify the resources aggregation scheme in accordance to their needs; it is also not possible to dynamically modify the resources number on the basis of the real system workload, in the name of saving energy and to achieve a more efficient and sustainable environment. But given the flexibility in resources management through the Cloud Computing paradigm, it seems a promising approach to provide flexible Grid Computing infrastructures through the combination of the Grid and Cloud paradigms.

Hence the idea to develop GaaS (Grid as a Service), a work started two years ago in order to find a solution to make distributed computing environments, based on Grid model, more "elastic" and more sustainable, exploiting the features of the Cloud model.

We assume as Grid reference architecture the one implemented by the middleware gLite-EMI, developed in the context of EGI (European Grid Infrastructure). The gLite-EMI middleware provides a Grid infrastructure that is accessible to community members organized into Virtual Organizations (VO). The infrastructure provides users with high level services for scheduling and running computational jobs, accessing and moving data, and obtaining information on the infrastructure itself. Those provided are services for authentication/authorization (e.g. VOMS), resources allocation and discovery (e.g. LB/WMS), infrastructure Information System (IS). Computing resources (WNs) are provided by means of CE (Computing Element) that is an endpoint with a set of queues handled by an LRMS (Local Resource Management System). User can access these services from a User Interface (UI).

The outcome of our work is to provide an usage model already familiar to Grid users, with the enhancement of infrastructure flexibility usually provided by Clouds. Since the provided service is a Grid environment, tailored to the user needs, and integrated in a general preexisting Grid infrastructure, our model can be classified as a Platform-as-a-Service system applied to Grid environments. GaaS actually consists of the following four services:

1. to add new computing resources (Worker Node Service: **GaaS WNS**);
2. to aggregate existing computing resource in a new queue (Queue Service: **GaaS QS**);
3. to add a new grid site for an existing VO (Grid Site Service: **GaaS GSS**);
4. to create a suited runtime environment for a set of applications on existing or new computing resources (Application Environment Service: **GaaS AES**).

The **GaaS WNS** service manages the integration of new WNs to an existing Computing Element queue. The system acquires cloud resources (public or private), configures resources as grid WN select the queue where the WN has to be inserted and ask for a CE local resource management system conguration. This service is useful in these scenarios: to realize efficient datacenters management in term of energy consumtion reduction and to provide users with more resources if they are not locally available.

The **GaaS QS** service manages the integration of a new queue. In this case, the system ask for a CE configuration, specifying the set of underlying WNs and the queue policies. A possible use case for this service is a community who needs particular queue policy (i.e. queue priority for the community users, different values for max walltime of execution, etc.).

The **GaaS GSS** service manages the integration of a new grid site for an existing VO in an existing grid infrastructure. The system acquires cloud resources (public or private), ask for GaaS WNS and GaaS QS services, configures an Information System and select the grid infrastructure where new grid site has to be inserterted into. A possible use case for this service is related to a community which has to share resources for the life time of a project. An other ideal scenario is

related to the need to build on the scratch, and for a short time period, a tailored grid testbed.

The **GaaS AES** service performs the Application Enviroment Service. The system deploys a suitable combination of GaaS WNS, GaaS QS and GaaS GSS services and it provides, on allocated resource, all applicative software stack according to users community requirements. The software stack can be builded on the basis of software portfolio. A possible use case for this service is related to a community which wishes add to the benefits of all the other listed above services also the chance of a proper *"applicative middleware"*.

All GaaS services has been developed by means of a prototype integrated into the context of the S.Co.P.E. Datatcenter at University of Naples Federico II. The S.Co.P.E. Datacenter consts of up to two thousands computational cores. Computational resources are embedded in a grid-based context that provides the user with high level services for scheduling and running jobs, accessing and moving data, and obtaining information on the infrastructure state. This prototype is based on the gLite-EMI Grid middleware and on the Open-Nebula Cloud management system. The modularity of OpenNebula allows for fast introduction of new features to the management system, hence, it allows the easy integration of the Cloud-provided resources into gLite.

A subset of the S.Co.P.E. worker nodes are assigned to the OpenNebula managed resources pool. Such worker nodes host an hypervisor, i.e., Xen, to create virtual machines (VM), that are then used as dynamically provided resources for the Grid infrastructure. VMs are used to both create Grid's worker nodes and management services such as E. The main efforts in the prototype development were (i) the extensions of gLite related to the configuration repositories, in order to dynamically add resources (e.g., sites, queues, worker nodes), and (ii) the enabling of their fast provisioning.

In particular, as regard as the latter issue, dynamically provided resources have to be homogeneous with the Grid Worker Nodes in terms of software configuration (e.g., operating system, application software, etc.). To this end, we prepared a VM template, i.e., a virtual disk configured with the needed operating system and software, compliant to gLite, that is duplicated many times as needed. In many IaaS management systems (and in OpenNebula as well), VM templates are usually stored in a "templates repository". A new VM is created copying the selected template to the running location of the VM. The duration of this process is the main factor in the resources deployment time. Since the copy process involves the storage infrastructures that are hosting the template repository and the newly created VMs disks, assuming that a minimal VM template is several hundreds of megabytes big, the process, for each VM creation, takes a time in the order of dozens of seconds. The actual time it takes is strictly connected to the underlying infrastructure and to the effective dimension of the copied template, but, in any case, there are several hundreds of megabytes copied around in the infrastructure.

To optimize the infrastructural resources used during the provisioning process, and to reduce the overall provisioning time, we took into account the peculiarities of the GaaS system. In particular, we made the following observations:

1. all the VMs are mandated to host the same operating system and software configuration, i.e., there is just one VM template;
2. the VMs operational destinations (e.g., WN, CE, etc.) are differentiated among them by a few configuration differences in respect to the VM template;
3. most of the operations performed on the newly created VMs are reads, since writes are usually performed on a dedicated network storage provided by the Grid Infrastructure.

We are currently working on solutions able to allow new communities wishing to use the grid to instantiate new grid infrastructure also for the non existing VOs. Even if the presented work is a successful proof-of-concept, many issues still have to be solved. In particular the we have to assess the applicability of virtualized resources in HPC contexts, the payed overhead, and the possibility to extend the model to a mix of virtualized and physical resources according to the users needs. Moreover, we are also planning an evaluation of the impact on management operations and costs of our approach, in order to integrate a smart management of resources with the aim of providing energy savings, in a Green Computing perspective.