# Programmable Grids Framework Enabling QoS in an OGSA Context

John Soldatos[1], Lazaros Polymenakos[1], and George Kormentzas[2]

[1] Athens Information Tchnology
19.5 Km, Markopoulou Ave., Peania GR-19002, P.O Box: 68, Greece
{jsol,lcp}@ait.gr
[2] University of the Aegean
Karlovasi, Samos, 83200, Greece
gkorm@aegean.gr

**Abstract.** Configuring, monitoring and guaranteeing service levels is a key prerequisite towards supporting the wave of emerging Grid computing applications, especially those relating to e-business. Managing QoS across a Grid infrastructure is particularly complex since QoS must be addressed at three different levels: network, middleware and application. Programmable Grids extend programmable networks concepts to Grid infrastructures and can alleviate this complexity. In this paper we elaborate on the importance of network, middleware and application level QoS, and accordingly provide a framework for designing and implementing programmable APIs that can handle QoS. The framework leverages the structure and properties of OGSA services, while exploiting functionality of OGSI containers. Moreover, early implementation experiences with this framework and Globus (GT3) are reported.

## 1 Introduction

Grid computing constitutes a distributed computing paradigm enabling virtual organizations (VO) to share resources towards achieving their goals [1]. Existing deployments have demonstrated benefits for various user groups. The emphasis is nowadays on the standardization and the commercialization of the Grid towards accelerating its development and adoption. The Global Grid Forum (GGF) has taken a significant standardization step through the initial specification of the Open Grid Services Architecture (OGSA) [2], which achieves a standardized way for virtualizing and accessing resources.

OGSA boosts the commercialization of the Grid through supporting applications that deliver improved business results [3], [7]. The commercialization of the Grid poses however new requirements and associated technical challenges. Enterprise applications demand guaranteed service levels. As a result Grid Service Level Agreements (GSLA), along with mechanisms, tools and techniques for managing them, need to be introduced [4]. Managing GSLAs can be greatly facilitated through a programmability layer on top of a Grid. This gives rise to grid architectures consisting of dynamic, reconfigurable on-demand and highly customizable computing, storage and networking environments, called programmable Grids. Programmable Grids have their roots in the programmable and active networks paradigms.

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3038, pp. 195–201, 2004.

The purpose of this paper is to introduce an infrastructure enabling programmable operations across a Grid. We view these operations in a programmability context (i.e. based on the IEEE P1520 initiative [5]) rather than in an active networks context. We describe a framework for offering open programmable APIs (Application Programming Interfaces) on Grid systems towards automatically provisioning and managing QoS. The notion of QoS and related metrics is familiar to the networking community, since it is an integral component of broadband networks. Nevertheless, Grid QoS is not confined to networking QoS, but extends to middleware and application level QoS [6]. Our framework for programmable operations covers all levels and focuses on resource management. Other issues (e.g., security management for pre-authorized and authenticated QoS services) are out of the scope of this paper.

Early research work has already produced results regarding mechanisms for pooling and using resources (e.g., [8]), as well as for defining and negotiating generalized GSLAs (e.g., [10]). With respect to QoS, other works have come up with mature general-purpose architectures allowing management of diverse Grid resources (e.g., [9]). The most characteristic example is the GARA architecture, which defines APIs for manipulating reservations of different resources in uniform ways. Our design focuses on an infrastructure for open, programmable OGSA based APIs that can be employed to implement GLSA models and related resource management policies.

The rest of the paper is structured as follows: Section 2 discusses the main requirements from a Grid QoS programmable framework. Section 3, presents our framework for exporting programmable OGSA based APIs. Section 4, reports on work conducted using the Globus Toolkit version 3 (GT3), towards implementing certain aspects of this framework. Finally, section 5 concludes the paper.

## 2 Programmable Grids for QoS

### 2.1 Network QoS

State of the art networking infrastructures provide immense capacity capable of supporting emerging Gird applications. Future broadband networks are expected to provide improved support for larger scale Grid infrastructures. Programmable Grids can provide the means for leveraging the capabilities of both existing and emerging networking infrastructures. The programmable infrastructure should virtualize the networking resources and provide a set of interfaces for accessing and managing these resources. Network resources and metrics of interest include bandwidth, throughput, as well as delay and loss figures. Emphasis is put on the capacity, utilization and availability statistics on particular links, interfaces and routing paths (e.g., [6]).

Exposing programmable interfaces for accessing and managing network resources, hinges on the communication with entities that can provide this information (i.e. information providers), while also allowing execution of configuration operations. Such entities are element layer management (ELM) systems, network layer management (NLM) systems, network resource managers (NRM), bandwidth brokers (BB), as well as directory systems (e.g., LDAP directories). The scope of these systems depends on the network topology, the technologies deployed, as well as on administrative policies. Usually NLM, BB and NRM systems handle a whole administrative domain.

Programmable interfaces must be exported through the Grid middleware layer (e.g., Globus). In OGSA terms, this implies that operations accessing network resources and statistics should be available as Grid Services within an OGSI [12] compliant container, (e.g., GT3). As a result, an OGSA based programmable Grids framework delivers network control and management functionality as a set of Grid Services. This delivery can leverage recent work on network management operations based on Web Services (e.g., [13]).

### 2.2 Middleware QoS

Middleware layer QoS deals with the lifecyle and execution of Grid services, and the related management and allocation of IT resources (i.e. servers, storage). Grid middleware such as the Globus Toolkit provides the infrastructure for managing middleware QoS. In particular, Grid middleware platforms provide the means for allocating virtualized resources to data and/or compute centric tasks.

An OGSA based programmable framework must export APIs for managing allocation of CPU and storage resources to the Grid Services. GT3 provides a set of management mechanisms enabling job submission on distributed resources, namely the Globus Resource Allocation Manager (GRAM). GRAM defines and implements APIs and protocols allowing Grid clients to securely instantiate jobs, according to the functionality of remote schedulers. GRAM constitutes a readily available resource management mechanism, which however can be enriched with more advanced resource management features (e.g., [11]). GRAM is used in conjunction with a wide range of other schedulers (e.g., the Sun 's Grid Engine (SGE), IBM's LoadLeveler, or the Portable Batch System (PBS) [14]). Instead of using GRAM, a new programmable API could be defined and implemented.

### 2.3 Application QoS

Application QoS deals with the performance of the operations entailed in the Grid application. In an OGSA context a Grid application consists of a set of Grid services, which interact to achieve an application specific goal. Accomplishing this goal includes: (a) configuration of a grid service, (b) data transfer to a grid service (from a calling service), (c) execution of a remote service, (d) data transfer from the grid services (back to the calling service). QoS for each of these operations depends on the middleware and network QoS. As a result an API for managing application level QoS, hinges on network and middleware QoS capabilities.

## 3 Programmable OGSA Grid Architecture

Taking into account the requirements outlined in the previous section, we devised an OGSA based framework for delivering programmable APIs for QoS management. Figure 1, depicts a programmable framework for managing network layer QoS across a loosely coupled heterogeneous Grid infrastructure. From a network perspective the various devices of the Grid are clustered into distinct administrative domains. Each domain consists of several devices (e.g., routers, switches, homogeneous clusters), which are controlled by a single administrative entity. In the scope of each administrative domain, we envisage a network model and resource management structure resembling current networking infrastructures. In particular, networking resources of the administrative domain are controlled through a centralized or distributed NMS. A distributed NMS will most likely rely on ELM systems to collect management information from the network elements. QoS parameters relating to inter-domain SLAs are controlled by Bandwidth Brokers, which is perfectly in line with the IETF DiffServ model.

The programmable framework accesses NMS, ELM and BB functionality based on a set of OGSA Grid Services. These services are capable of taking and combining raw network measurements, as well as of configuring network parameters. Delivering such Grid Services hinges on the following steps:

1. Selecting operations of interest, as well as the network information pertaining to these operations.
2. Defining the Grid services corresponding to the network resource management operations of interest. This step hinges the specification of the programmable APIs for network QoS. This specification entails a degree of abstraction, since the API is defined independently of particular domains and systems (e.g., NMSs, BBs).
3. Specifying the network resource parameters and metrics involved in the Grid API operations (e.g., capacity, utilization, delay). These parameters will form the major part ServiceData of the OGSA Grid Services.
4. Implementing methods (in the form of low-level drivers) for accessing the respective operations of the NMS, ELM and BB systems. The OGSA based implementation of these methods entails an OGSI middleware platform (e.g., GT3), as well as a container where such services can be deployed.

The implementation produces an API for controlling Network QoS parameters. This API will be independent of the peculiarities of the network resource management systems of the domains engaged in the Grid implementation. The virtualization layer on top of these systems allows translation between operations and parameters specific to the domains, to domain and system independent operations and parameters. Specifically, the Grid Services API (API Programmability Layer in Figure I), constitutes the domain independent implementation of an API for network QoS control. The `Service Data` defined in the scope of these Grid Services forms the domain independent Grid parameters.

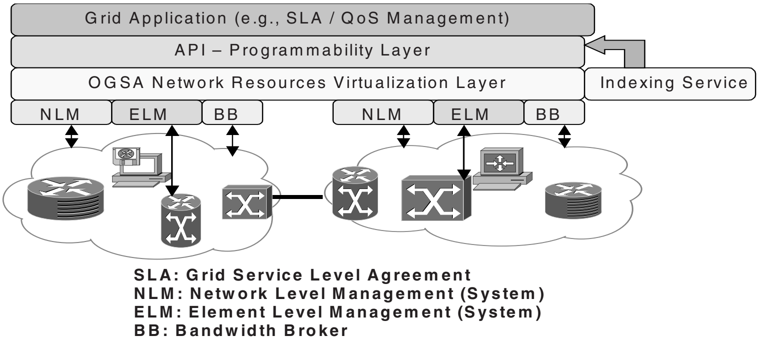
Grid Application (e.g., SLA / QoS Management)
API – Programmability Layer
OGSA Network Resources Virtualization Layer
Indexing Service
NLM
ELM
BB
NLM
ELM
BB
SLA: Grid Service Level Agreement
NLM: Network Level Management (System)
ELM: Element Level Management (System)
BB: Bandwidth Broker

**Fig. 1.** Programmable Framework for Network QoS Management

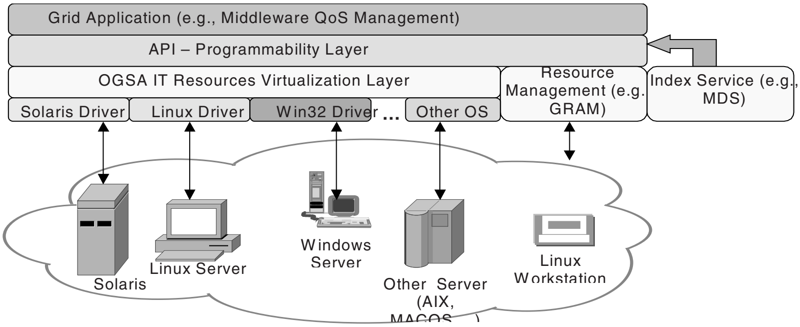
Grid Application (e.g., Middleware QoS Management)
API – Programmability Layer
OGSA IT Resources Virtualization Layer
Resource Management (e.g. GRAM)
Index Service (e.g., MDS)
Solaris Driver
Linux Driver
Win32 Driver
...
Other OS
Solaris
Linux Server
Windows Server
Other Server (AIX, MACOS, ...)
Linux Workstation

**Fig. 2.** Programmable Framework for Middleware QoS Management

The set of OGSA services for configuring, monitoring and managing network QoS parameters can be registered with an information indexing service (e.g., the Globus Monitoring and Discovery Service (MDS)). Accordingly, Grid applications can query the indexing service based on `Service Data` parameters.

The framework for managing Middleware QoS (Figure2) reveals many similarities to the framework for Network QoS control. In particular, it aims at delivering functionality for accessing QoS information about computing resources (e.g., storage, CPU utilization), and accordingly processing this information towards scheduling jobs. This implies that methods for accessing low level capabilities of the Grid machines (e.g., the operating systems, processors) are available. Thus, interfacing with system level services relating to the Grid donors is required. This depends on implementing drivers for interfacing with each one of the Grid constituents. As a result, the steps towards delivering OGSA Grid Services as a set of programmable APIs involve:

1. Specifying the parameters engaged in the job scheduling and infrastructure resources management operations.
2. Defining these parameters as OGSA `ServiceData`, and design Grid Services that access and configure those resources.
3. Implementing methods for accessing and altering the information specified within `Service Data`, for all of the platforms (e.g., Linux, Windows, Sun Solaris, IBM AIX, Mac OS) participating in the Grid.

Grid services relating to middleware QoS can also be registered with an Index Service (e.g., GT3 MDS) to allow scheduling applications to query for services based on `Service Data` parameters.

Using APIs for middleware QoS control and resource management, advanced scheduling mechanisms can be implemented. Commercial Grid platforms, as well as Globus provide readily available resource management and job scheduling functionality. Nevertheless, the proposed approach can still be followed towards enhancing existing functionality, or even implementing totally new schedulers (e.g., customized for particular Grid applications).

Auditing, controlling and provisioning applications' QoS is a matter of appropriately managing the underlying operation impacting network and middleware QoS. Hence, APIs for controlling application QoS can be delivered as shown in Figure 3. Application QoS APIs will be implemented as sets of other API calls concerning network and middleware QoS. These APIs will be used in the scope of

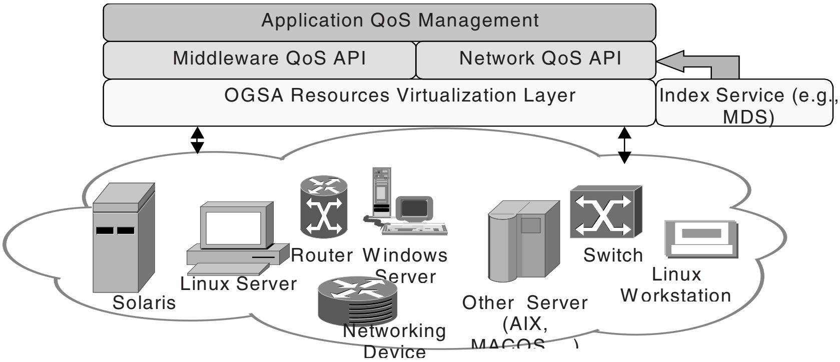
Application QoS Management
Middleware QoS API
Network QoS API
OGSA Resources Virtualization Layer
Index Service (e.g., MDS)
Solaris
Linux Server
Router
Windows Server
Networking Device
Other Server (AIX, MACOS, ...)
Switch
Linux Workstation

**Fig. 3.** Programmable Framework for Grid Application QoS Administration

resource management applications defining and setting QoS levels for Grid applications. Thus, they constitute a valuable interface for Grid applications administration.

## 4 OGSA Implementation Experience

As a first step, we have implemented low level drivers accessing network management systems and system level functionality. Implementation of such drivers has been up to date limited to a single administrative domain containing three machines equipped with GT3.

As far as network level QoS is concerned, OGSA Grid Services accessing networking parameters through interfacing with an ELM system have been implemented. This ELM system executes SNMP operations on SNMP enabled network element [13]. The ELM system exports a Web Services interface, which facilitates the implementation of OGSA Grid Services. Based on Grid Services exposing network management functionality we have implemented simple OGSA clients querying and configuring element level network management information.

At the middelware layer we have implemented drivers allowing Grid services to access metrics and statistics regarding computing resources (i.e. CPU in particular). These metrics have been used to enrich the `Service Data` of OGSA services. Towards interfacing (Java) GT3 OGSI services to system level functions pertaining to our Solaris machines, we relied on the Java Native Interface (JNI). Accordingly, we implemented simple job schedulers (i.e. load balancers based on processor utilization metrics). This implementation provided a proof of concept regarding the lower layer of our programmable middleware QoS framework. As part of ongoing and future work, we intend to implement all aspects of the framework.

## 5 Conclusions

Future Grid infrastructures must guarantee QoS levels. QoS is not only concerned with the networking infrastructure, but also with the middleware and application structure. Programmable Grids can leverage the OGSA towards designing and implementing programmable APIs for QoS control. OGSA imposes a structure for delivering a programmable framework, while OGSI application containers provide essential supporting features. In this paper we have presented a framework for structuring, designing and implementing OGSA compliant programmable functionality for QoS management. Early experimentation with GT3 has demonstrated several of concepts of the framework. We expect future work to provide a more complete underpinning.

## References

1. I. Foster, C. Kesselman, S. Tuecke, 'The Anatomy of the Grid: Enabling Scalable Virtual Organizations', International Journal of Supercomputer Applications, 15(3), 2001.
2. I. Foster, D. Gannon, H. Kishimoto (editors), 'The Open Grid Services Architecture', Global Grid Forum Draft, draft-ggf-ogsa-ogsa-011, September 23, 2003.
3. I. Foster, D. Gannon, H. Kishimoto, Jeffrin J. Von Reich, 'Open Grid Services Architecture Use Cases', GWD-C (draft-ggf-ogsa-usecase-1.7), September 18, 2003.
4. Akhil Sahai, Sven Graupner, Vijay Machiraju, Aad van Moorsel, 'Specifying and Monitoring Guarantees in Commercial Grids through SLA', Hewlett-Packard Laboratories, White Paper, November 2002.
5. Biswas, J., Lazar, A. et al "The IEEE P1520 Standards Initiative for Programmable Network Interfaces", IEEE Communications Magazine, pp. 64-71, October 1998.
6. G-QoSM: Grid Service Discovery Using QoS Properties, Al-Ali RJ, Rana OF, Walker DW, Jha SK, Sohails S, Computing and Informatics Journal, Special Issue on Grid Computing, Institute of Informatics, Slovak Academy of Sciences, Slovakia, 2002, 21(4), pp. 363-382.
7. J.Soldatos and Lazaros Polymenakos, 'Enterprise Grids: Drivers and Implications', Athens Information Technology Report, December 2003.
8. Jacek Gomoluch and Michael Schroeder, 'Market-based Resource Allocation for Grid Computing: A Model and Simulation' in the Proc. of the 1st International Workshop on Middleware for Grid Computing, Rio de Janeiro, Brazil, 2003.
9. I. Foster, C. Kesselman, C. Lee, B Lindell, K. Nahrstedt, A. Roy, "A Distributed Resource Management Architecture that Supports Advance Reservation and Co-Allocation", in the Proceedings of the International Workshop on QoS, pp. 27-36, 1999.
10. Czajkowski, K., Foster, I., Kesselman, C., Sander, V., and Tuecke, S., "SNAP: A Protocol for Negotiating Service Level Agreements and Coordinating Resource Management in Distributed Systems", 8th Workshop on Job Scheduling Strategies for Parallel Processing, Edinburgh, Scotland, July 2002.
11. K. Czajkowski, I. Foster, and C. Kesselman, 'Co-allocation services for computational grids'. In the Proceedings of the 8th IEEE Symposium on High Performance Distributed Computing. IEEE Computer Society Press, 1999.
12. S. Tuecke, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maquire, T. Sandholm, D. Snelling, P. Vanderbilt, 'Open Grid Services Infrastructure (OGSI), Version 1.0' (draft), GWD-R (draft-ggf-ogsi- gridservice-29), April 5, 2003.
13. Dimitris Alexopoulos and John Soldatos, 'Open Programmable XML based Network Management: Architecture and Prototype Instantiation', submitted for publication in the IEEE Communications Magazine, special issue on XML based Management of Networks and Services, under review.
14. Luis Ferreira et. al. 'Introduction to Grid Computing with Globus', IBM redpaper, December 2002.