RT-Grid: A QoS Oriented Service Grid Framework*

Hai Jin, Hanhua Chen, Minghu Zhang, and Deqing Zou

Cluster and Grid Computing Lab,
Huazhong University of Science and Technology, Wuhan, 430074, China
hjin@hust.edu.cn

Abstract. Effective and efficient *Quality of Service* (QoS) management is critical for a service grid to meet the requirements of both grid users and service providers. We incorporate QoS management into *Open Grid Services Architecture* (OGSA) and provide a high-level middleware to build complex applications with QoS guarantees. We introduce a scalable framework of information service in which capabilities of services are described and advertised as metadata of grid services. Based on that, a QoS policy based dynamic service scheduling strategy is proposed. The experiment result on the prototype proves the availability of our framework.

1 Introduction

OGSA presents a vision of an integrated approach to support both e-science and e-business [1]. The most striking technical contributions of OGSA to the Globus are in the area of extensibility and manageability. As implementation of OGSA, *Open Grid Service Infrastructure* (OGSI) [2], such as Globus Toolkit 3.0, and *Web Service Resource Framework* (WSRF) [3], such as Globus Tookit 3.9, provide the basic functionalities without QoS management.

In a highly competitive service grid environment, quality of service is one of the substantial aspects for differentiating between similar service providers. It is important for the grid platform to support dynamic service allocation in accordance with QoS policy. We provide a platform for service designers from specific organizations of business domains to publish the business semantics document described with standard interface description languages and QoS metadata schema. Different service providers implement the service semantics, deploy their services and register their service implementations with different *QoS metadata*. The semantic information and QoS metadata are managed and advertised by information service, and they are used for service selection and scheduling.

2 QoS-Based Service Grid Framework

Service-oriented view simplifies service virtualization through encapsulation of diverse implementations behind a common interface that standardizes the business function.

* This paper is supported by National Science Foundation under grant 60273076 and 90412010, ChinaGrid project from Ministry of Education, and the National 973 Key Basic Research Program under grant No.2003CB317003.

K.-M. Leiw et al. (Eds.): PDCAT 2004, LNCS 3320, pp. 345–348, 2004.

Our grid environment provides a set of *virtual service* (VS) to users.

$$Grid = \{VS\}$$

We develop a tool for service designer to define and advertise a *VS* by defining service interfaces as gwsdl porttypes (defined in OGSI specification). We call an implementation of *VS* a *physical service* (PS). Each *VS* is a set of redundant *PS*s.

$$VS = \{PS\}$$

All the *PS*s of the same *VS* have the uniform *operation* (*O*) set and are invoked in the same manner, while even the diverse implementations of the same operation by different *PS*s may have different *QoS Costs* (C).

$$PS = \{<O, C>\}$$

Here, the 2-tuple $<O, C>$ defines an operation with a QoS cost expression which may include the resource description and other capacity metadata for this operation.

We designed a XML-schema (in Fig.1) for operation cost. The schema is used as metadata schema of a service. Using the schema service providers can define the cost of any operation in the service separately. Based on the service QoS metadata schema, different policies can be designed for service scheduling.

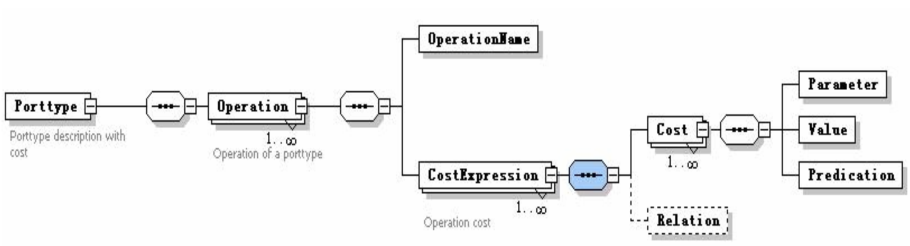

Fig. 1. XML Schema for the Service QoS MetaData

A user's request to the grid in the policy is described as following:

$$Request = VS_{request}<o_i, c_{request}>$$

A domain server is responsible for autonomous intra-domain administration and inter-domain interaction. It consists of a *Domain Service Broker* (DSB), a DIS and a *QoS Manager* (QoSM). The architecture of a domain server is shown in Fig.2.

When there comes a request, DSB looks up the DIS or GIS, selects the proper service and negotiates an SLA according to the QoS metadata of the service and the requirements of the incoming request $VS_{request}<o_i, c_{request}>$. A proxy instance is created to dispatch and monitor the coming communication. The QoSM is responsible for the QoS of network and SLAs management. The SLAs status is monitored and maintained by QoSM during the life cycle of the grid service instance. It also allows reasoning about the status of SLAs related to an application context across multiple administrative domains by contacting and querying QoSM in other domain servers.

Our grid information service architecture aims at supporting service discovery. The architecture of the GIS is like distributed *Universal Description, Discovery and Integration* (UDDI). It is a distributed system with peer-to-peer [4][5] structure. DISs and GISs are organized in hierarchical scheme. DIS updates the service information to specify group information server dynamically. When a grid customer wants to access a service, it first looks up local DIS containing the service information, and continues to search the GIS upon failure.

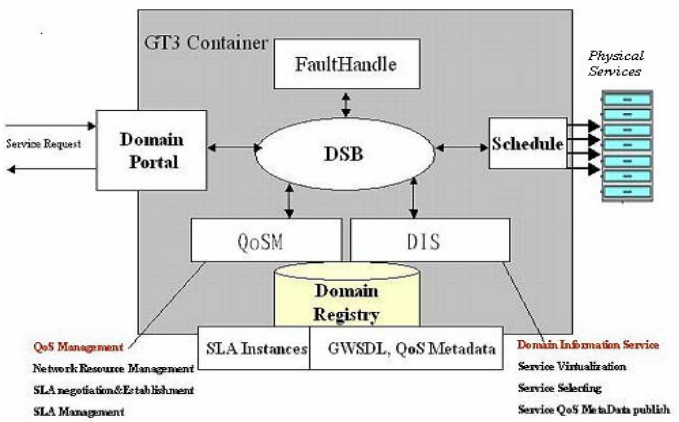

Fig. 2. Domain Server Architecture

3 Experiment Results

Our service grid prototype uses a virtual calculator service as an example, with implementations from three different domains of the Cluster and Grid Computing Lab of Huazhong University of Science and Technology. Services deployed on heterogeneous sites provide different response capacity and network cost. On site 211.69.206.128 an IBM x235 server runs, 211.69.206.199 an HP rx2600 server, and 202.114.14.252 a PC server equipped with an Intel Xeon 1GHz CPU and 512MB memory. We have three groups of experiments. In each group, 20 random recodes of requests, which have the same time constrain, are selected. The deadlines specified in the three groups are 240 ms, 260 ms, and 450ms, respectively. According to the metadata of the services all the requests are allocated to a site. Response time of each access is shown in Table.1.

The marked data shows the cases that the real-time constrains are not met. The preliminary results demonstrate that the strategies have a good role on our service grid prototype. We also implement a fault-tolerant strategy. If a fault occurs during the access, the request can be re-scheduled to another service with higher response capacity.

4 Conclusions and Future Works

We extend the service abstraction in OGSA for QoS management and provide a high-level middleware to build complex application with QoS management. QoS-based service virtualization framework is proposed. In next steps, we will enhance the QoS policy based service-scheduling model to make it support more algorithms, analyze the cost of grid middleware and try to improve the performance.

Table 1. Experiment Result: Resoponse Time

No	211.69.206.128	211.69.206.199	202.114.14.252
1	220	260	301
2	200	250	310
3	220	251	310
4	221	240	280
5	230	541	270
6	230	260	290
7	235	201	270
8	211	210	280
9	411	190	290
10	220	211	300
11	220	230	280
12	221	255	250
13	190	190	251
14	235	201	270
15	200	210	280
16	220	200	260
17	210	210	270
18	220	180	340
19	211	190	270
20	221	200	260

References

[1] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke, "The Physiology of the Grid An Open Grid Services Architecture for Distributed Systems Integration", *DRAFT document of Globus Project of the University of Chicago*, February 17, 2002.

[2] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, D. Snelling, and P. Vanderbilt, *Open Grid Services Infrastructure*, February 17, 2003.

[3] K. Czajkowski, D. F. Ferguson, I. Foster, J. Frey, S. Graham, I. Sedukhin, D. Snelling, S. Tuecke, and W. Vambenepe, "The WS-Resource Framework", *http://www.globus. org/wsrf/*

[4] M. Ripeanu, "Peer-to-Peer architecture case study: Gnutella network", *Proceedings of International Conference on Peer-to-peer Computing*, 2001.

[5] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications", *ACM SIGCOMM*, 2001.