
Book Section:
Haji, M.H., Djemame, K. and Dew, P.M. (2006) Deployment and performance evaluation of a SNAP-based resource broker on the White Rose grid. In: Proceedings of the International Conference on Information & Communication Technologies: From Theory to Applications. 2006. (Damascus, Syria, April 2006). IEEE, pp. 3365-3370. ISBN: 0-7803-9521-2.

Deployment and Performance Evaluation of a SNAP-based Resource Broker on the White Rose Grid

Mohammed. H. Haji Karim. Djemame Peter. M. Dew

Informatics Research Laboratory
School of Computing
University of Leeds, UK
Email :{mhh, karim, dew}@comp.leeds.ac.uk

Abstract

Resource brokering is an essential component in building effective Grid systems. The aim of this paper is to evaluate the performance of a SNAP (Service Negotiation and Acquisition Protocol) based resource broker on a large distributed Grid infrastructure, the White Rose Grid. The broker uses a three-phase commit protocol to reserve resources on demand, as the traditional advance reservation facilities cannot cater for such needs due to the prior time that it requires to schedule reservations.

Experiments are designed and carried out on the White Rose Grid. The experimental results show that the inclusion of the three-phase commit protocol provides a performance enhancement on a large distributed Grid Infrastructure, in terms of the time taken from submission of user requirements until a job begins execution. The results support those previously obtained through the use of mathematical modelling and simulation. The broker is a viable contender for use in future Grid resource brokering implementations.

1. Introduction

Grid computing is an approach to distributed computing, which has the potential to provide users with high performance resources in a seamless virtual organisation [2, 3]. To support application execution in the context of the Grid, it is desirable to have a resource brokering service [9]. In [4] the authors discussed a simple SNAP (Service Negotiation and Acquisition Protocol) based resource broker and a more sophisticated SNAP broker, following a three-phase commit protocol. SNAP [13] is an appropriate choice in the design and implementation of a user-centric broker, since it provides the means to negotiate and acquire resources that meet the user's application requirements through Service Level Agreements (SLA). The broker focuses on applications that require resources on demand such as the UK e-Science DAME (Distributed Aircraft Maintenance Environment) project [1], which is a joint project between the Universities of Leeds, York, Sheffield and Oxford, and Rolls Royce and Data Systems & Solutions as industrial partners. The use of traditional advance reservation cannot cater for such needs, due to the time required to schedule the reservation. The resources must be secured immediately prior to run-time. Therefore the SNAP broker also incorporates a three-phase commit protocol that provides services to ensure decisions are made with up-to-date information, resources are differentiated and the nominated resources are secure before jobs are submitted.

Experimental results obtained through mathematical modelling and simulation have shown that the use of the three-phase commit SNAP broker provides a performance enhancement over a simple SNAP broker, in terms of the time it takes for a job to be successfully submitted and begin execution [4]. However the issue of how well the three-phase commit SNAP broker compared to the simple version performs over a large distributed Grid infrastructure has not been addressed and is the subject of this paper. An example of a large infrastructure is the White Rose Grid (WRG) which consists of high performance computing resources at Leeds, Sheffield and York Universities [14]. The WRG exhibits heterogeneous resources and spans over multiple administrative domains, providing an environment which differs from previous studies [4] due to its true nature as a Grid infrastructure. Experiments are carried out and performance results show that the inclusion of the three-phase commit protocol provides a performance enhancement in terms of the time taken from submission of user requirements until a job begins execution.

The paper is organised as follows. Section 2 provides an overview of the architecture of the SNAP broker and a description of the three-phase commit protocol to secure resources. This is followed, in section 3, by a description of the White Rose Grid. Section 4 presents and discusses the experimental results. In section 5, the paper ends with conclusions and a discussion of future work.

0-7803-9521-2/06/$20.00 ©2006 IEEE.

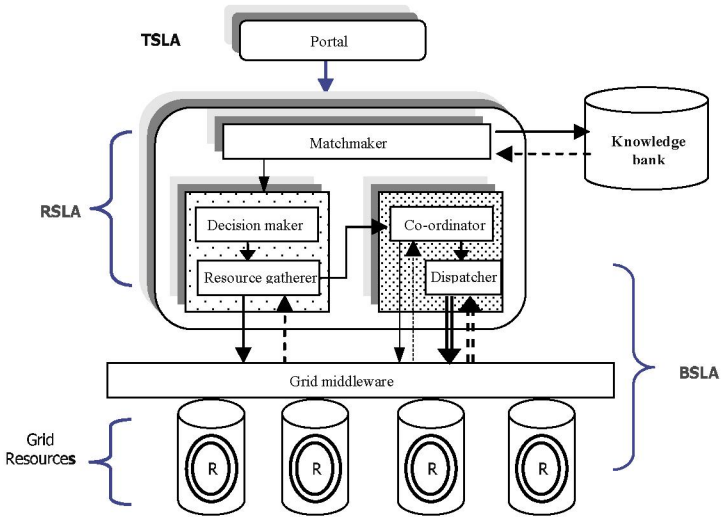

Figure 1: Grid Resource Broker Architecture within the SNAP Framework.

2. SNAP-based Resource Broker

2.1 Architecture

The broker's architecture (Figure 1) shows the components that comprise the broker. In this architecture, the broker begins by parsing the user requirements submitted through a Grid portal; this is the first layer of SNAP Task Service Level Agreement (TSLA) [4]. The second layer, Resource Service Level Agreement (RSLA), uses a *Matchmaker,* supplied with the parsed user requirements, to contact a Knowledge Bank (KB). The latter is a repository that stores static information on all resources. The broker can access this information on behalf of the user for each resource he/she is entitled to use. The information stored in the KB as attributes include the number of CPUs, the operating system, memory, storage capacity and past behaviour performance of a resource. Relating to the latter, resources are classified as low/high priority according to whether they meet a pre-defined level of performance such as reliability. Referring to Figure 1, on receiving the information the Matchmaker forwards the details to the *Decision Maker* that evaluates the information and categorises the potential resources into categories. This corresponds to their significance, i.e. that some resources are reliable and valuable while others are acceptable.

The *Resource Gatherer* based on the information received from the Decision maker queries the information provider on each selected resource to gather dynamic information on their status.

Once all queries have reported back to the broker the information is forwarded to the *Co-ordinator*, which nominates the resources to handle the tasks and secures them for utilisation through the use of immediate reservation.

Once the resources are secured, the final procedure, which is part of the Binding Service Level Agreement (BSLA), is executed by the *Dispatcher* by submitting the task and binding it to the resources. In the remainder of the paper, a broker following this protocol, without the additional enhancements discussed in section 2.2, is referred to as a simple SNAP broker.

2.2 Three-Phase Commit Protocol

In [4], the problem of oscillation that could occur between the broker and the information provider without a successful job submission was highlighted. The key issue outlined is that information obtained from the information provider for each resource may be out-of-date by the time a decision is made as to where the job should run and the Co-ordinator proceeds to attempt to reserve the chosen resources. This problem arises, since the broker does not know if the status of a resource changes until it re-contacts its information provider. An efficient solution will be to receive a signal from each resource if its status changes rather than needing to contact the information provider on each chosen resource.

The proposed protocol follows the three phases of the simple SNAP broker, indicated in Section 2.1. However, the first phase is strengthened by the addition of *probes,* which are software sensors, to enable rapid updates of changes in status to the resources that are under consideration for use. Specifically, when the Resource Gatherer queries the information provider on candidate resources, it simultaneously transmits probes to the resources, thereby entering into the first phase of the commitment. The purpose of the probes is to enable the broker to be kept updated while waiting for all queries to return at their various times. This helps to reduce the likelihood of the oscillation situation, as it provides a constant vision for the broker of the resources' status. This allows the broker to remain up-to-date while the information provider reports back. The approach of having the information provider broadcast resource status to the probes listening to any changes is more efficient than having the broker repeatedly contacting the information provider for updates after the initial contact.

Once all information provider queries have reported back to the broker and updates from the probes are

acknowledged, the information is forwarded to the *Co-ordinator*, which executes the second phase of the commitment, by nominating resources to handle the task. It then informs the probes associated with the nominees to request the resources' information provider to evolve into a state which indicates to other users that though the resource is not active, it is unavailable. On such a request the information provider would reserve the resource and present an indication to any candidate interested in its use that it has entered a transition phase.

Once the resources are secured through the use of immediate reservation, the third and final phase is executed by the *Dispatcher*. This phase binds the task to the resources and their information provider signals to any incoming client that the resources are active and have committed.

3. White Rose Grid

The White Rose Grid (WRG) is a virtual organisation comprising of three universities: the Universities of Leeds, York and Sheffield [14]. The WRG is heterogeneous in terms of its underlying hardware and operating system. Two large compute nodes are situated at Leeds (Maxima and Snowdon), one at York (Pascali) and another at Sheffield (Titania). The specification of these compute nodes is described below:

- *Snowdon*, a Beowulf 256 CPU running at 2.2GHz and 2.4Ghz Intel Xeon processors.
- *Maxima*, Sun Fire 6800 server (20 Ultrasparc 3Cu, 44GB memory, 100GB Storage), 5 Sun V880 servers and 2 TB storage.
- *Pascali*, Sun Fire 6800 server (20 Ultrasparc 3 Cu, 44GB memory, 100GB Storage), 1 Sun V880 server and 1TB storage.
- *Titania*, 10 Sun Fire V880 Servers (8xUltrasparc Cu 900MHz, 32GB) and 2TB Storage.

The WRG middleware infrastructure is enabled through the use of Globus 2.4 and 3.0.2, while Sun Grid Engine (SGE) [15] handles job scheduling.

4. Experiments on the WRG

4.1 Overview and Objectives

As shown previously in [4] the three-phase commit SNAP broker provides a performance enhancement over the simple SNAP broker in terms of the time interval between submission (to the broker) of user requirements and the job beginning execution. However this was only carried out through the use of mathematical modelling and simulation. The issues of how well the three-phase commit SNAP broker compared to the simple version performs over a large distributed Grid infrastructure such as the WRG has not been addressed and is the subject of this study.

The experiments primarily use resources from all four machines across the three sites. This is to investigate whether the resource status provided by the probes used in the three-phase commit protocol still provide an enhancement by ensuring that decisions are made on the basis of up-to-date information. Specifically, the experiments involve a comparison of the performance of the simple compared to the three-phase commit SNAP broker. The experiments presented in this section are designed on the basis of the following objectives:

- To show that the simple SNAP broker and the three-phase commit protocol have been successfully deployed on the WRG.
- To investigate the behaviour of the three-phase commit SNAP broker over a large distributed Grid infrastructure when scenarios occur in which a performance enhancement over the simple SNAP broker is expected.

A further complimentary experiment to the above is carried out that combines both WRG and local Grid testbed resources. The latter consists of 8 machines each with a Pentium IV processor (1.2 GHz), 256MB RAM running Linux 2.4 and Globus 2.4, and connected via a fast (100 Mbps) Ethernet. This is to ensure the brokers could cope with two different environments simultaneously.

4.2 Deployment of the Three-phase Commit Broker on to the WRG

The information provider, Monitoring and Directory Service [17] (MDS) is configured differently from that of the default installation in order for the GRIS (Grid Resource Information Service) running on each machine to display dynamic queue information. GRAM (Grid Resource Allocation Manager) is recompiled and installed after the poling service was modified for updates from every 20 seconds to 1 second. This ensures the various stages of a job process from Pending, Active to Done is recorded as they occur. The Globus configuration (MDS, GRIS and GRAM) is repeated across the three sites following the same procedures. It is worth noting that local scheduling

queues are the gateway to resources. Each queue activates a Prolog and Epilog script (supported by SGE), when a job starts and ends respectively. This is to inform the server associated with each queue of the resources status and is used by the probes to gain their updates. Currently the information provider MDS provides information on request and does not broadcast resources status to the users. The server is developed to enhance the current system by providing this facility, i.e. resources status broadcast.

4.3 Experimental Design

The experiments presented here can be described in terms of two scenarios:

1. **Scenario1**: The resources appropriate for the job are taken and the broker must wait until they become free before submitting the job. This experiment was carried out on the WRG only.
2. **Scenario2:** While the broker is in the process of making a decision as to where the job should be submitted, another job is submitted (see below for more detailed description). This was carried out with the use of both the WRG and the local Grid testbed.

The first scenario provides a setting in which the effectiveness of the probes in providing a vision of the resources can be investigated on a large distributed Grid infrastructure. Specifically, a user's job to be submitted requires 4 resources. Each broker is given access to these resources and on each of these, another job is running for a fixed duration. Both brokers are considered in turn.

Two experiments are performed, based on this scenario. In the first experiment, an additional job that is submitted to make the resources unavailable has a fixed duration of 40 seconds. This job is submitted immediately before the broker is executed. The information provider response time (i.e. the GRIS response time) is then varied between 30 and 360 seconds. This is to reflect the fact on a universal Grid infrastructure spanning across several countries it may be time consuming to obtain information from the MDS for some resources. Additionally, this time may vary considerably depending on, for example, the number of users concurrently accessing the same GRIS [7, 8] and the load on the machines.

The response time was varied by adding a variable delay into the code. The time taken between the broker beginning execution and the broker becoming aware that the resources are free is then recorded, in addition to the time taken before the user's job begins execution and the number of information provider contacts made.

In the second experiment based on scenario 1, there is no artificial delay in the information provider response time. Instead, the duration for which the resources are unavailable, which was previously fixed at 40 seconds, is varied between 40 and 360 seconds.

For the simple SNAP broker, as soon as the information provider informs the broker that the resources are free, the time is noted and stored. For the three-phase commit SNAP broker, the time is stored when a signal has been obtained from all 4 resources that they are free.

The experiment based on scenario 2 is used to investigate the three-phase commit SNAP broker's performance when resources are initially free but their status changes during co-allocation. In this experiment the broker has access to 12 resources (from both the WRG and the local Grid testbed). This time the broker requires three resources. Note that if a resource is taken during co-allocation, the simple SNAP broker only becomes aware of this when it re-confirms with the resources. In that case it must repeat the process of gathering the status of resources and co-allocating the user's job. In order to highlight the scenario whereby the broker is required to repeatedly contact the information provider and attempt to co-allocate the user's job, additional jobs are submitted at time intervals chosen to coincide with each attempt at co-allocation that the simple SNAP broker makes. Additionally, the resources taken are chosen to be the highest priority available so that there is always a conflict between the additional jobs and the broker. This experiment is used to determine whether the vision of the resources that the probes used in the three-phase commit protocol do indeed enable the broker to obtain fast enough updates to decrease the likelihood of oscillation between broker and resources.

4.4 Experimental Results

4.4.1 Scenario 1

The results for the first experiment relating to scenario 1 (Figure 2) show the time taken between the broker beginning execution and the user's job beginning execution as a function of the information provider response time. Both brokers became aware the resources were freed 15 seconds prior to that shown in Figure 2, for each time interval. Further, for the simple broker the number of information provider contacts remained constant at two after 40 seconds, whereas for the three-phase broker only one contact to

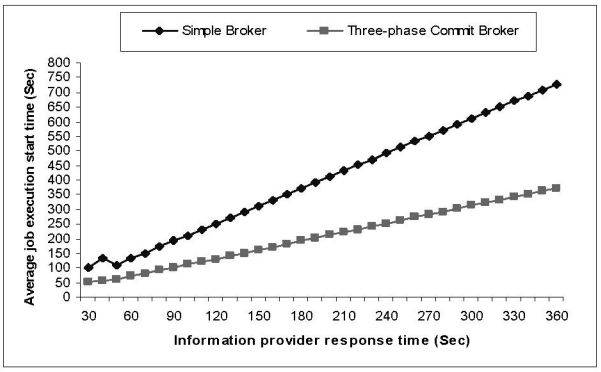

Figure 2: Time taken for job to begin execution as a function of information provider response time.

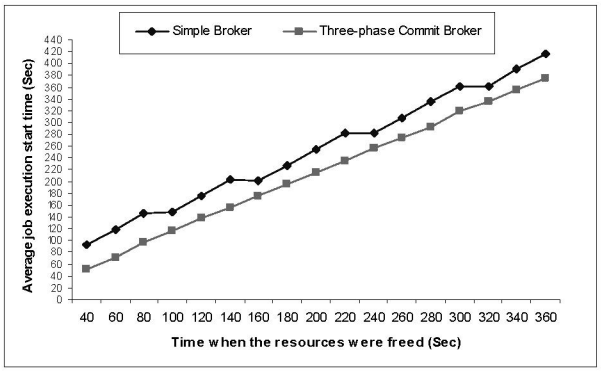

Figure 3: The time taken for the job to begin execution as a function of the time for which resources are unavailable.

the information provider is recorded throughout the experiment.

The three-phase commit SNAP broker begins the execution of the user's job much sooner than the simple SNAP broker as a consequence of the use of the probes, providing an average performance improvement of 48%. The three-phase commit protocol provides the broker with updates of the resource status much faster than having to repeatedly contact the information provider after the initial contact. Usually, the longer the response time, the longer it takes before either broker is aware that the resources are free. For the three-phase commit SNAP broker, there is only one contact with the information provider, hence increasing the response time has little effect until it exceeds the 40 second period for which the resources are taken. For the simple SNAP broker, the effect is apparent even for faster response times. However the time taken before this broker becomes aware of the change in resource status is shorter when the response time is 50 seconds than when the response time is 40 seconds. This is due to the fact that when the response time is 40 seconds, the simple SNAP broker needs to contact the information provider three times before it is aware of the change in status, while if the response time is 50 seconds, only two contacts are required.

The results for the second experiment relating to scenario 1 are shown in Figures 3. Figure 3 shows the time taken until the user's job begins execution, which the three-phase commit protocol provides an average performance improvement of 18%.. The results are given as a function of the time for which the resources are unavailable. As shown in Figure 3, the simple broker performance is related to repeatedly contacting the information provider to find out when the resources are released, and not having the facility of the probes to provide rapid updates. Further, third party jobs may end at varying times at the three sites, and consequently this results in the information provider not acknowledging the jobs completion before it re-contacts the resources again. This also effects the start time of some of the user's jobs. For example, Figure 3 shows the same job execution start time (140 sec) for two different times when the resources are freed (80 and 100 sec respectively).

4.4.2 Scenario 2

Figure 4 show the results for the experiment carried out in relation to scenario 2. Figure 4 shows the three-phase commit SNAP broker takes just over 50 seconds to submit and begin execution of the user's job, irrespective of the number of other third party jobs submitted. Since the probes ensure rapid updates of the status of the resources, the three-phase commit SNAP broker is aware that a resource has been taken very quickly after it occurs and consequently chooses an alternative resource, providing an average performance improvement of 75%. It successfully submits the user's job before any other resources are taken. However, the simple SNAP broker takes longer when more resources are taken, since it is unable to identify changes in resource status fast enough to successfully submit the user's job before more resources are taken.

4.4.3 Overall Results

The experiments discussed above have demonstrated that the three-phase commit protocol ensures that the broker has access to fast updates on the status of resources. This has enabled a performance enhancement in a number of specific scenarios and

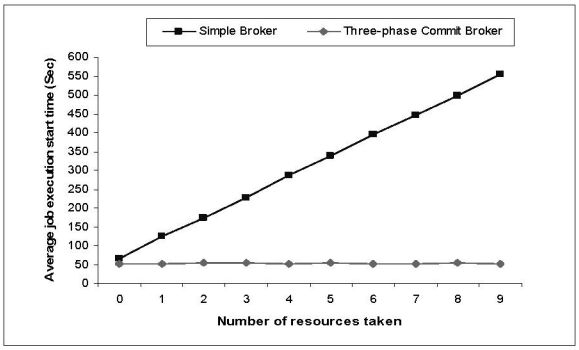

Figure 4: Time taken to begin job execution as a function of number of additional jobs submitted.

with an average performance enhancement of up to 75%.

The experiments discussed illustrate the value of using the three-phase commit protocol. This has been achieved by considering specific scenarios, where the vision of resources provided by the use of probes enables faster submission of a job than would otherwise be possible.

Studies have also been carried out to investigate when the three-phase commit broker will perform weakly. In the worst-case scenario (unfavourable to the three-phase commit protocol) where resources required by the broker are idle and there is no competition for their use the three-phase commit protocol will always perform just as well as the simple broker. The simple broker will not outperform the three-phase protocol as the protocol follows the same procedure as explained in Section 2.2, with the first phase strengthened. Further there is no overhead cost associated with setting up the probes since this occurs concurrently when contacting the MDS.

5. Conclusion

This paper presents a resource broker deployed on the WRG which has been developed through the use of the SNAP framework. The broker differentiates the resources that are capable of handling the user's job. The three-phase commit protocol was developed with the knowledge that other users potentially require the same resource. This is why probes are used to keep a vision of any changes, resources are secured before submission and binding of the tasks to the resource. The user is insulated from having to keep a log of the resources that he is entitled to use or the specific details on how the Grid mechanisms operate.

It has been shown that the SNAP-based resource broker is a viable contender for use in future Grid implementations. This is supported through mathematical modelling and simulation [4] and with further experiments on a large distributed Grid infrastructure (WRG).

An important step for further exploration of these results is to evaluate the three-phase commit protocol using resources from many countries spanning over different continents. Also the systematic comparison of the three-phase commit protocol with existing methods found in the literature such as that used in AppLeS [8].

Acknowledgement

This project is part-funded by a Collaborative Research and Development grant under the DTI Technology Programme. Further information can be found at: www.dti.gov.uk/technologyprogramme.

References

[1] J. Austin et al. Predictive Maintenance: Distributed Aircraft Engine Diagnostics. In Grid2: Blueprint for a New Computing Infrastructure. I Foster, C. Kesselman (Eds.), Chapter 5, Morgan Kaufmann, 2003, 2nd edition

[2] I. Foster, C. Kesselman, S. Tuecke, The Anatomy of the Grid: Enabling Scalable Virtual Organizations, International J. Supercomputer Applications, 15(3), 2001.

[3] The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. I. Foster, C. Kesselman, J. Nick, S. Tuecke, Open Grid Service Infrastructure WG, Global Grid Forum, June 2002.

[4] I. Gourlay, M. Haji, K. Djemame, P.M. Dew. Performance Evaluation of a SNAP based Grid Resource Broker. In Proceedings of FORTE'2004 Workshops (1st European Performance Engineering Workshop), M. Nunez, A. Maamar, F.L. Pelayo, K. Pousttchi and F. Rubio (Eds.), Toledo, Spain, September 2004, Lecture Notes in Computer Science 3236, pp. 220-232, Springer.

[7] X. Zhang, J. L. Freschl, J. M. Schopf. A Performance Study of Monitoring and Information Services for Distributed Systems. In Proceedings of the 12th International Symposium on High-Performance Distributed Computing, Seattle, Washington, June 2003, pp. 270-281.

[8] H. N. Lim Choi Keung, J. R. D. Dyson, S. A. Jarvis, G.R.Nudd. The Globus Monitoring and Discovery Service (MDS-2): A Performance Analysis. In Proceedings of the 19th UK Performance Engineering Workshop (UKPEW'2003), S. Jarvis (Ed.), Warwick, UK, July 2003, pp. 103-116

[9] K. Krauter, R. Buyya, and M. Maheswaran. A Taxonomy and Survey of Grid Resource Management Systems. International Journal of Software: Practice and Experience, Vol. 32, No. 2, Wiley Press, USA, February 2002.

[10] F. Berman and R. Wolski. The AppLeS Project: A status report. Proceeding of the 8th NEC Research Symposium Berlin, Germany, May 1997

[12] J.M. Schopf. A General Architecture for Scheduling on the Grid. Technical Report, Argonne National Laboratory ANL/MCS-P1000-10002, 2002

[13] K. Czajkowski, I. Foster, C. Kesselman, V. Sander, S. Tuecke. SNAP: A Protocol for Negotiating Service Level Agreements and Coordinating Resource Management in Distributed Systems. In Proceedings of the 8th Workshop on Job Scheduling Strategies for Parallel Processing, Edinburgh, Scotland, July 2002.

[14] White Rose University Consortium. http://www.wrgrid.org.uk.

[15] Sun Microsystems. Sun Grid Engine, http://www.sun.com/software/gridware 2004

[16] SLA Management in a Service Orientedb Architecture. K. Djemame, M. Haji and J. Padgett. In Proceedings of ICCSA'2005, Singapore, May 2005, Lecture Notes in Computer Science 3483, pp.1282-1291.

[17] S. Fitzgerald, I. Foster, C. Kesselman, G. von Laszewski, W. Smith, S. Tuecke. "A Directory Service for Configuring High-Performance Distributed Computations". Proc. 6th IEEE Symposium on High-Performance Distributed Computing, pp. 365-375, 1997.