Developing Web Services in a Computational Grid Environment

Xiaobo Yang, Mark Hayes, Andrew Usher
Cambridge eScience Centre
University of Cambridge
Wilberforce Road
Cambridge CB3 0WA, UK
{xy216, mah1002, acju2}@cam.ac.uk

Mark Spivack
Department of Applied Mathematics and Theoretical Physics
University of Cambridge
Wilberforce Road
Cambridge CB3 0WA, UK
M.Spivack@damtp.cam.ac.uk

Abstract

Grid and Web services are both hot topics today. In this paper, we will present some ongoing work and planned future work at the Cambridge eScience Centre. After an introduction to these technologies in the context of Grid applications development, we describe two use-cases: a database of results in computational fluid dynamics (CFD) and a small computational Grid for aircraft engineering design. As Grid services are moving towards Web services, we continue to make use of the Globus Toolkit v2.4 (GT2.4), without adopting the Open Grid Services Architecture (OGSA) wholesale. In our scenario, GT2.4 integrates distributed computing resources including HPC and clusters while Web services wrap the scientific code as a service.

1. Introduction

Grid and Web services are currently attracting intense interest. As Web services mature towards an industry standard, they have received a great deal of attention from the Grid computing community. Extensions of Web services have been developed as Grid services, which are defined in the Open Grid Services Infrastructure (OGSI) [14] specifications. Therefore, a Grid service is actually a Web service that conforms to a particular set of conventions besides those for Web services. For example, both Grid services and Web services are defined in terms of standard WSDL (Web Services Definition Language) and make use of standard Web service binding technologies such as SOAP (Simple Object Access Protocol). But Grid services conventions are beyond these and attempt to address problems such as the state of services with lifetime management. The relationship between Grid services and Web services is given in detail by Grimshaw and Tuecke [11].

The Open Grid Services Architecture (OGSA) attempts to define a set of high level Grid services for batch job management, monitoring, database access etc. OGSI was intended to provide a conceptual foundation for building these higher level services which are used directly by Grid applications. One can illustrate this framework in a classic layered stack diagram (see Fig. 1).

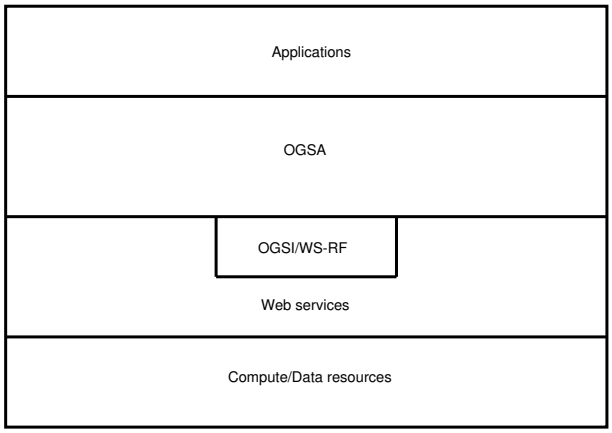

Figure 1. OGSA stack.

While the OGSI specifications provide a way to develop Grids based on Web services standards, Foster noted three major concerns at the GlobusWORLD 2004 conference: OGSI places "too much stuff in one specification", "does not work well with existing Web services tooling" and is "too 'object oriented' ". The new WS-Resource Framework (WS-RF) [1] proposed by the Globus Alliance, IBM, HP and others addresses these issues. Future implementations of the Globus Toolkit will be based on WS-RF rather than OGSI. At the time of writing, the WS-RF specifications have just entered the standards process at OASIS [2]. Stable implementations may be some time in appearing. Therefore we chose to continue using the "mature" GT2.4 in our test bed. In this paper, we will first describe some preliminary work on Web services, followed by our thoughts on developing secure Web services in a Grid environment. Our planned future work will be described with conclusions given in the last section.

2. Preliminary work on Web services development

Grids [10] offer the possibility of aggregating the capabilities of geographically distributed computing resources. In today's engineering and scientific research, numerical computational techniques are heavily utilised. Usually these techniques are in constant pursuit of ever larger computing resources in order to produce increasingly precise and reasonable results. Also research work today commonly involves large scale cooperation between many (international) institutes and universities. While the Grid aims to share resources and set up virtual organisations (VOs), Web services help to realise this by exposing applications, databases and other resources as network-accessible components.

One of the projects based at the Cambridge eScience Centre [3] is the application of Grid-based technology to combustion CFD [4]. With the computational Grid environment setup successfully [12, 15], some meaningful computational results are ready to be shared with the CFD community. Therefore this project is selected as our preliminary case study. First, we set up an XML database with descriptions of each dataset, or meta-data. Information such as the creator of the dataset, the date the dataset is created together with some physical parameters is included. With the database set up, we next want to make it visible to the public. Therefore we develop a Web service to visit the database. We expose this service with Apache Axis [5] and provide a client as a Java servlet. Now everyone is able to consume our Web service though the servlet, i.e., he or she can query the database to search for results of interest. The WSDL file of the service is also published so that users are able to develop their own clients. The structure of the service is depicted in Fig. 2.

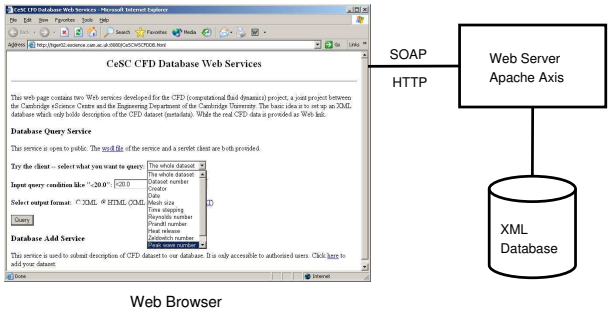

Figure 2. Structure of the CFD query Web service.

Part of a query result of the above CFD query Web service is show in Fig. 3. The query condition is set as querying "Peak wave number" with the condition "< 20.0" through a Web form shown in Fig. 2. A query statement is automatically constructed within the Java servlet, then submitted to the CFD Query Web service. The query result is handled by the Java servlet to generate a static HTML page using XSL Transformations (XSLT) [6] or simply return the query result in XML format.

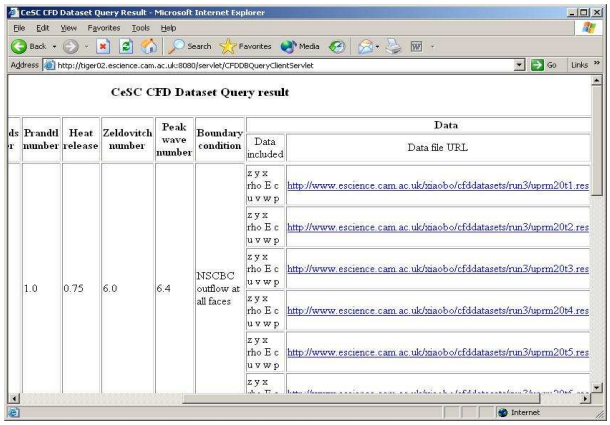

Figure 3. One query result of the CFD query Web service.

As we can see from Fig. 3, CFD datasets are given with descriptions while the real data is returned as Web link. This is mainly due to the huge size of each dataset. It also makes it flexible to let dataset providers control how to publish their datasets. For example, they may publish their data on a Web server, a FTP server, etc. Of course security policies may be applied by each dataset provider. In order to help dataset providers to publish their dataset descriptions (XML files to be added to our database), we have developed another Web service. Obviously we do not want everyone to

write to our database. Therefore security policies must be considered for this service. With mature transport-level security techniques (SSL/TLS over HTTP), we successfully tested basic HTTP authentication based on SSL. Apache Axis provides a powerful function which enables developers to develop their own handlers to deal with the incoming/outgoing SOAP messages. A handler has been developed to allow only secure connections to our service. Furthermore, message-level security has also been tested. In this case, we tested XML signature [7] to sign the document the user wants to add to the database to keep its integrity.

3. Developing secure Web services in a computational Grid environment

In a computational Grid environment, security is one of the first and most important factors considered as it often involves direct access to hardware, software or data on computing resources. Therefore when a service is deployed, it is necessary to apply some security policies to secure the service. As described above, standard transport-level security techniques (SSL/TLS over HTTP) are natural choices for end-to-end security and authentication. In a computational Grid environment, we also require authorisation policies to use a specified resource and single sign-on (SSO) for access to multiple resources. The Globus GSI (Grid Security Infrastructure) [9] provides these features. An example will be given below to describe the problem we meet when developing a Web service to make use of a computational Grid.

Suppose we want to wrap the job submission function of GT2.4 as a Web service, a proxy is required when a job is submitted to a GRAM (Globus Resource Allocation Manager). Basically we can provide such a proxy in two ways.

1) The Web service asks its client to provide the user's passphrase of his or her private certificate. Then a proxy will be created by the Web service itself.

2) The Web service asks its client to provide a proxy as one of its input parameters. In this way the client must have user's private certificate to generate such a proxy either by GT2.4 itself or MyProxy [13].

In the first method, transmission of user's passphrase of his/her private certificate and access to his/her private certificate by the Web service is needed. While in the second one, a user should keep his/her own private certificate with Globus Toolkit client or MyProxy client installed. The second method is widely adopted by portal developers since a user is better protected than the first way. Therefore we would also select this method to develop our Web services.

4. Future work

EMGrid [8], one of the projects hosted at the Cambridge eScience Centre, is the context for our second case study on developing secure Web services within a computational Grid environment. This project aims to optimise the design of shape and of material properties for aircraft by using radar scattering. The numerical simulation code splits naturally into two separate parts. The first part, say, code A, is used to provide raw data for further processing. The second part, code B is used to process the raw data generated by code A for visualisation. Code A, which is highly computationally intensive, usually runs on a backend high performance computing facility (HPCF). Code B is suitable for running on a normal Linux cluster.

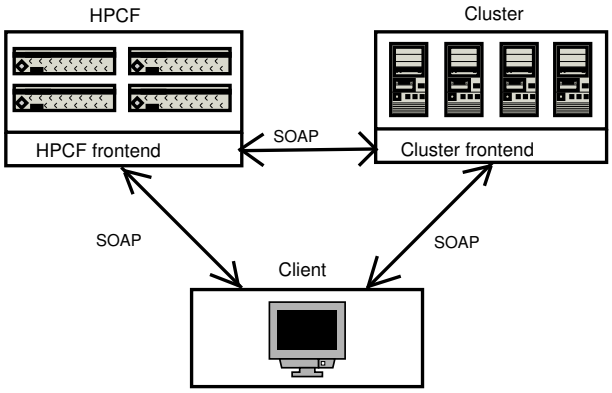

Figure 4. Architecture of a Web service system.

Figure 4 illustrates the architecture of such a Web services system. Two distinct backend hosts, a HPCF and a cluster, are included. Each backend host has its own frontend to expose its service. As mentioned above, the HPCF frontend contains the code A service (service A) while the cluster has code B service (service B). At the first stage, we hope to visit each service separately from a client. While at the second stage, we hope to combine both services to provide an integrated service for scientists or engineers to perform real-time design with feedback corrections. Although here we simply put a "Client" in Fig. 4, the real client can be as simple as a client java code of service A or B. It can also be as complex as a portal Web server (Fig. 5). According to our experience in portal and portlet design [15], a Web portal server could be a good choice to provide a user friendly interface for end users to consume services A and B.

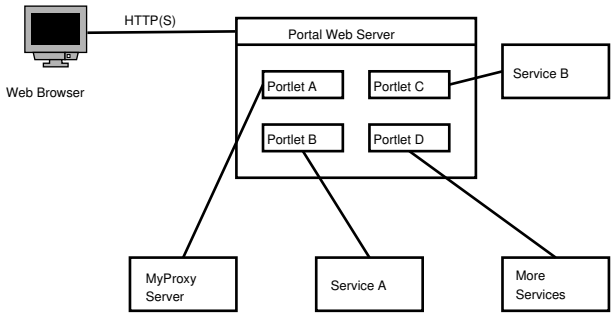

Figure 5. Complex client – portal Web server structure.

5. Conclusions

In this paper, we first give a description of our preliminary work developing Web services for Grid applications where secure authentication and authorization are required. Both transport-level security (HTTP+SSL/TLS) and message-level security (XML signature) have been tested successfully as authentication mechanisms. We have also proposed an architecture using Web services in a computational Grid environment, with Globus GSI as an authorisation framework.

References

[1] *http://www.globus.org/wsrf/*.
[2] *http://www.oasis-open.org/*.
[3] *http://www.escience.cam.ac.uk/*.
[4] *http://www.escience.cam.ac.uk/projects/cfd/*.
[5] *http://ws.apache.org/axis/*.
[6] *http://www.w3.org/TR/xslt*.
[7] *http://www.w3.org/TR/xmldsig-core/*.
[8] *http://www.escience.cam.ac.uk/projects/emgrid/*.
[9] I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke. A security architecture for computational Grids. In *Proc. 5th ACM Conference on Computer and Communications Security Conference*, pages 83–92, 1998.
[10] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *International J. Supercomputer Applications*, 15(3), 2001.
[11] A. Grimshaw and S. Tuecke. Grid services extend Web services. *Web Services Journal*, 3(8):22–26, 2003.
[12] K. Jenkins, X. Yang, M. Hayes, and S. Cant. Distributed computational fluid dynamics. In *Proc. UK e-Science All Hands Meeting 2003, edt. S.J. Cox*, pages 862–864, Nottingham, UK, 2-4 September 2003.
[13] J. Novotny, S. Tuecke, and V. Welch. An online credential repository for the Grid: MyProxy. In *Proc. of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press*, 2001.
[14] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maguire, T. Sandholm, P. Vanderbilt, and D. Snelling. Open Grid Services Infrastructure (OGSI) Version 1.0. *Global Grid Forum Draft Recommendation, 6/27/2003*.
[15] X. Yang, M. Hayes, K. Jenkins, and S. Cant. The Cambridge CFD Grid portal for large-scale distributed CFD applications. In *International Conference on Computational Science 2004, eds. M. Bubak et al., Springer-Verlag, LNCS 3036*, pages 478–481, Kraków, Poland, 6-9 June 2004.