A GRID SERVICE COMPUTING ENVIRONMENT FOR SUPPLY CHAIN MANAGEMENT

Sam Chung and George A. Orriss
Computing & Software Systems, Institute of Technology, Univ. of WA Tacoma,
Box 358426 1900 Commerce St., Tacoma, WA 98402, USA

Keywords: Grid Services, Web Services, Service-Oriented Computing, Supply Chain Management, Business to Business (B2B) application integration.

Abstract: This paper proposes to develop a Grid Service Computing Environment for Supply Chain Management. Current research into Grid Services for distributed systems has resulted in interesting questions being raised as to whether or not the Open Grid Service Architecture can be applied to developing a Supply Chain Management system. If so, how will it affect development of SCM systems as a typical example of Business-to-Business (B2B) application integration? Since as much recent development has been focused on resource allocation in a Grid environment, the B2B application integration using Grid computing is still relatively unexplored. By developing a Supply Chain Management system using the Open Grid Service Architecture and the Globus toolkit, this research will provide an infrastructure for composing existing services into a system that can be utilized for Supply Chain Management. The result of this project is a Grid environment that provides efficient and effective service management of available Supply Chain Management services. Also, we address some of the inherent issues of dynamic binding and automation associated with B2B transactions, such as those surrounding security protocols, service lifecycle, and instance creation.

1 INTRODUCTION

This paper develops a Grid service-computing environment (Fox, 2003) for Supply Chain Management (SCM), which relates to any service that concerns delivery of goods, services, and information from the supplier to the customer. There are two primary motivations behind this:

- Show the benefits of applying Grid service computing to SCM applications in an efficient and effective manner.
- Illustrate the abilities of the Open Grid Service Architecture (OGSA), which is a well-defined set of basic interfaces concerning the discovery and invocation of Grid and Web services, to allow for dynamic creation of customized services using separate and distinct Web services.

Initially, Grid computing is distinctly related to building supercomputing environments through the sharing of computing resources (Blythe, 2002). There are problems inherent to designing a system of this nature. The main question: Can an OGSA be applied to develop a SCM system? Considerable research has been conducted regarding the OGSA in order to illustrate the applicability of using it to incorporate Web services from various sources into a specific, customizable service environment (Foster, 2001 and 2002). Through this approach, this project shows the benefits and advantages of utilizing the OGSA to compose a SCM system. First and foremost, the nature of Grid computing has traditionally lent itself to the area of resource allocation for computing operations, specifically along the lines of scientific research. In addition, as Grid computing evolution has largely occurred in the last five years, much work related to employing Grid architecture to Web service binding is theoretical in nature, with little actual implementation to use as reference.

Building on this issue then, the second question arises: If such a system can be built, how will it affect the development of SCM systems as a typical example of Business-to-Business (B2B) application integration? Of primary concern with any B2B application, inherent issues have always existed relating to various security protocols, service

Chung S. and A. Orriss G. (2006).
A GRID SERVICE COMPUTING ENVIRONMENT FOR SUPPLY CHAIN MANAGEMENT.
In *Proceedings of the Eighth International Conference on Enterprise Information Systems - SAIC*, pages 112-117
DOI: 10.5220/0002466001120117

lifecycle management, and transaction instance creations regarding a particular system. For each system/service a client wishes to access for SCM systems, they must address these issues relating to binding to that particular service. What the OGSA brings to developing such systems is a single, stable environment that allows the client to be concerned with one service instance creation that handles service binding and all related B2B transactions via the use of one security certificate.

To achieve the desired results, implementation of the Globus toolkit, which is based on the OGSA protocols, is utilized in conjunction with an example of SCM. Globus has been designed as an all-in-one application for establishing a Grid computing environment while following protocols defined by the Global Grid Forum (GGF) relating to numerous trust and security issues (Foster, 2002, Alonso, 2004). The toolkit employs a standard set of interfaces that automate location and binding to services, among other functions. Additionally, these interfaces allow for platform and language independence via their design, thus allowing the incorporation of services previously isolated due to these constraints.

As Information Technology (IT) industry trends regarding cross-enterprise B2B collaboration lend itself to the concept of "virtual organizations" (Foster, 2001, Alonso, 2004), it seems logical to utilize Grid computing to attain these goals. Therefore, the system employs two major components: the Open Grid Service Infrastructure (OGSI) that provides a grid environment via use of the Globus toolkit, and designed/developed SCM service(s) in order to provide the virtualization (composition), which is necessary to allow access to resources across multiple heterogeneous platforms. Through illustrating the possibilities of the resulting environment, a new infrastructure for B2B interactions is realized that allows for interoperability of services without concern for language or platform dependence, as well as inherent issues of trust and security.

2 SCM AND WEB SERVICES

There have been previous implementations of the OGSA that have focused on resource the area of resource allocation for scientific computing operations. Most have been tied to resource management and allocation (Fox, 2003). Developing a system for SCM using the OGSA has not yet been done, so work relating to similar systems of this type must be referenced for design and implementation examples. It is possible to run test implementations utilizing the OGSA in conjunction with an existing Weather Grid Service (Schneider, 2003) as an example of the possibilities available by this technology. This service utilizes an existing service that provides current weather conditions (temperature) based on a user-supplied zip code, and while is a good starting point, doesn't provide an example of a Grid service composed of more than one Web service.

Regarding the approach of using the OGSA in distributed system B2B integration, e-services have been designed based on the configuration of custom Grid services defined using WSDL port types (Foster, 2002). This is the type of approach that will be applied in developing this system, as this has been shown to be a reasonable and flexible implementation. Each service composition must address issues of trust and security among possible unrelated services that are interacting, so the Globus toolkit handles standard semantics for interactions such as these.

Important to understanding and implementing this new type of service composition will be the handling of services located and binding to them in order to compose the Grid service desired. Research into "portlet" (Gannon, 2003) design provides insight into a graphic interface that shows available services in a tabbed environment, which can then be used as the composition base. As this is more theory than functionality-based, problems that occur when incorporating the interface aspect of the Grid environment needed addressing.

3 APPROACH

In order to develop a Grid service computing environment for SCM, there are two primary components that were needed for the project: the Open Grid Service Infrastructure (OGSI) for the SCM using OGSA and the SCM example. The SCM example was designed for performance related to purchasing order placement, fulfilment and other possible B2B transactions: retailer, suppler, and manufacture grid services. (Due to the space limit, the retailer grid services are explained in detail here.)

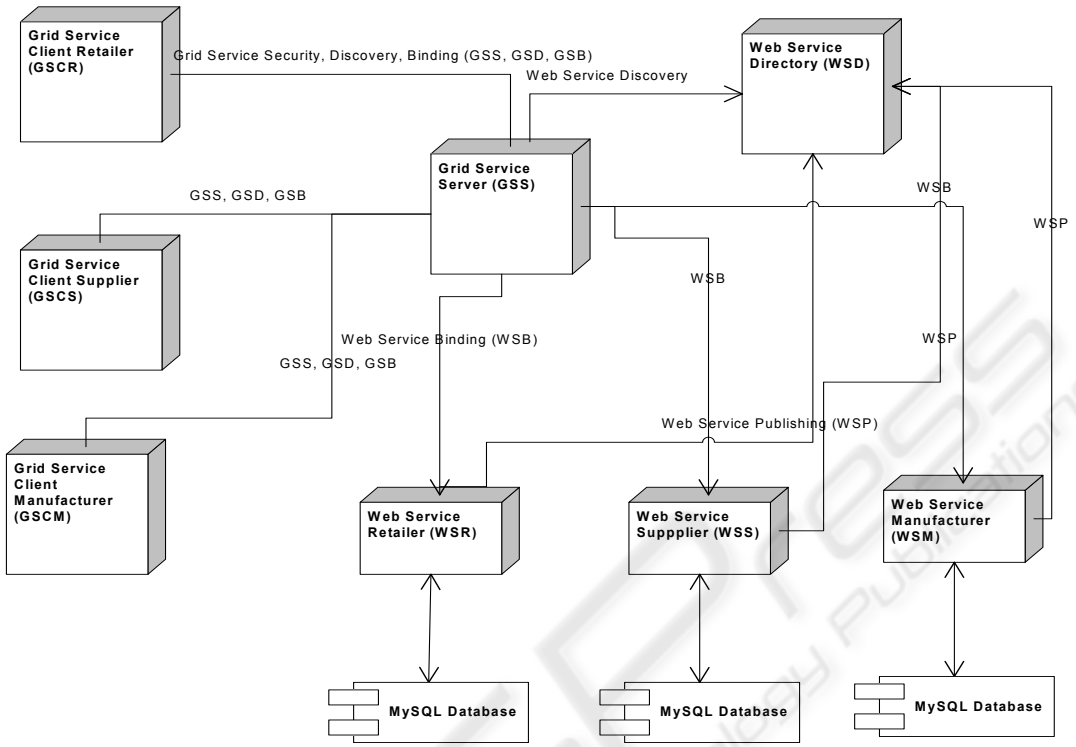

Figure 1: Overview of the Open Grid Service Infrastructure for Supply Chain Management.

3.1 OGSI for SCM

The OGSI for SCM is shown in Figure 1. The Grid Service Server (GSS) is the hosting environment (in this case, a Linux Fedora Core version 2 operating system is used.) There are three independent Web services shown in the deployment here, each providing services for one of the three actors such as a retailer, supplier, or manufacturer that would use the SCM system. Through manual discovery and composition by the Grid Service Administrator (GSA), necessary codes are supplied that handles the binding, composition, and deployment of these services into the GSS. Once this is completed, any Grid Service Client (GSC) can access services that they have permission to access. The GSA can provide any number of CA (Certificate of Authority)'s, depending on what types of services are to be provided to the clients utilizing that particular certificate. By utilizing this infrastructure, a client is concerned with one instance of service discovery, security and binding protocols, whereas a non-Grid approach would involve dealing with Web service's individual protocols in order to connect to each service. The OGSI provides a standardized security and management system that is controlled by the GSA, and allows ease of use for any number of clients.

3.2 Retailer Grid Service

For the implementation of this project, the Supply Chain Management system is designed with a Retailer being the first in the chain of execution. The Retailer in this example would be any of a number of stores wishing to order more inventory of soda for their locations. In order to do so, access to an available supplier's inventory is essential.

Implemented Solution: For purposes of the Grid service for retailer functionality, a fairly straightforward approach was taken, developing a Java-based Web service that could then be deployed in the Globus environment. The example used was that of a retailer (client) wishing to update inventory through ordering from a main supplier (warehouse). In order to achieve this, the application was designed with classes utilizing different operations relating to SCM functions.

Primarily, a user interface was the main focus here, providing a user-friendly environment for selecting different items to order, and prompting the user for the desired quantity. In order to assist in this process, the service populates a list of current inventory from the warehouse, so that the retailer can ascertain what is available. Each item is then added to a pending order along with the input quantity desired. Once the retailer confirms the order, the final order is displayed in a corresponding pane, a purchase order generated, and a corresponding database insertion of the purchase order occurs.

Once the transaction has been completed and the purchase order stored, it can then be accessed by the warehouse (supplier) for purposes of confirming/shipping the order. Using two essential files for deployment, 'deploy-server.wsdd' and 'build.xml,' the designed Web service is converted and launched as a Grid service in the Globus environment. Discussion of the important elements code in these files is covered here, but applies to any deployed Grid service.

Grid Service Methods: The Retailer Grid service consists of several methods, due to the variety of operations performed by the service. The main classes that comprise the service: The main classes are the RetailerGUI and DatabaseConnector with other classes playing important parts in the service functionality as well. Regarding the RetailerGUI class, the obvious primary functionality concerns the user interface, but it is important to note a very specific method that relates solely to Grid service functionality, and is unique from standard Java Web services. The getRetailerServiceInterface is a unique method that is designed to create and instance of the RetailerPortType necessary for communication with the Grid service running in a host container. For each client accessing a particular Grid service, a unique port is opened, and this connection is managed by Globus for all aspects of security, lifecycle, data transfer, etc.

The DatabaseConnector class features key methods to handle the connection to a remote database, in this example the MySQL system. Most of the remaining operations in this class relate to database information storage and retrieval, as methods such as *selectAllInventory, insertPO, updateDB,* and others perform. This is the class where much revision can be performed when it is desired to add additional functionality to the Retailer Grid service.

Also it should be mentioned that the Retailer interface and the RetailerImpl classes are key elements to the Grid service, as they provide essential functionality for utilizing the RetailerPortType.

4 RESULTS

The following sections discuss the results and lessons learned from the development of a supply chain management system using Grid service computing environment. The first two sections discuss the results obtained when deploying the Grid Service Computing Environment and the Grid services deployed in that environment, in order to answer the question as to whether or not an Open Grid Service Architecture can be applied to developing a Supply Chain Management system. The last section compares and contrasts Web and Grid Service Oriented Architectures in order to ascertain the feasibility of development of SCM systems as a typical example of Business-to-Business application integration.

4.1 Grid Services for SCM

Although the three components of this SCM system such as a retailer, a supplier, and a manufacturer performed different functionality, the design and implementation of all of them were very similar with regards to both positive and negative experiences. Because extensive works were done during the course of this research project attempting to use existing SCM systems, the actual design of Web services for composition into a Grid service proved to be more easily attained. By working with different code examples, it was possible to better understand Web services themselves and in doing so, understand how the Grid service concept is very similar to Web services, but allows one to deploy a variety of services that can be managed with one set of standards, as opposed to many. For clarity, Table 1 shows a comparison of the original Web service components and their corresponding components after deployment as Grid services.

Each of the above service components was originally designed as a Java-based Web service accessible through one or more user interfaces. By applying the OGSA to deploy the separate components as Grid services, they are transitioned into the corresponding Grid service. These are known as factory services in the OGSA, and as each

is deployed, the Globus toolkit allows any number of clients to connect to one or more services provided they have the correct CA installed on their machine.

What was clearly obvious through the work done with this deployed system was that it was possible to create numerous instances of the deployed Grid services and perform the designed functions without any problems due to security, speed, or reliability. The built-in management tools that are part of Globus allowed these tests to be successfully executed. The aspects of service management concerning security, instance creation, lifecycle, and the like were performed as "behind the scenes". As each service could run independently of the others and in numerous instances, it greatly illustrated the desirability of this type of environment for hosting and providing services to any number of clients with specific needs.

Table 1: SCM service comparison.

	Web Service	Grid Services
Retailer	Retailer Web Service/ RetailerGUI	Retailer FactoryService
Supplier (Warehouse)	Supplier Web Service/ SupplierGUI, SupplierShipGUI	Supplier FactoryService
Manufacturer	Manufacturer Web Service/ ManufacturerGUI	Manufacturer FactoryService

On the downside, the limitations mentioned earlier regarding the toolkit led to a good deal of time lost trying to get existing examples to work with the Globus toolkit that in theory should be possible. Although it could be cited as a shortcoming of the Globus toolkit, it is actually further illustration of the need for some type of centralized or common ground for providing these types of services with a minimal amount of overhead for the client.

4.2 WSOA vs. GSOA

Primarily, Web Service Oriented Architecture (WSOA) and Grid Service Oriented Architecture (GSOA) are the same conceptually, but the results show the GSOA to be better in terms of stability, reliability, and flexibility for incorporating different components.

As Table 2 shows, the GSOA, although lacking a defined registry for service location, offers many other features that current WSOA standards do not offer. To explain further, it is important to discuss some of the aspects in Table 2.

Client Connectivity: Client connectivity in a WSOA requires a client to deal with various connection protocols depending on how each separate service was composed. In contrast, the GSOA requires one standard protocol for connection. This is in concert with the **Security** feature, which requires a Certificate of Authority that allows the client to access all of the services of a particular Grid server. Any client wishing access simply needs to have the certificate installed relating to the set of services to which access is desired.

Table 2: WSOA vs. GSOA.

Feature	WSOA	GSOA
Client Connectivity	Multiple Protocols	Single Protocol
Instance Management	Separate and inconsistent	Single, consistent management
Security	Multiple Protocols	One Certificate (CA)
Binding to services	Various, confusing for user	Handled by GSA
Service design and deployment	Platform, language dependent	Platform, language independent
Service location	UDDI Registry	No current registry
Service composition	Not possible with differently built services	Any number of combinations

Instance Management: Each Web service must be specifically designed to handle its own management of instance creation, and many protocols don't address this effectively (Staab, 2003). This results in slower service response and possible unavailability. The Globus Toolkit uses default instance creation and management parameters that can be changed by the Grid Service Administrator to handle and allow multiple instances of a service to run at once, and killing off service instances if they remain inactive for the established period of time.

Binding to services: For each Web service a client wishes to connect to, there are issues of location, handshake protocol, and the requisite binding protocols that differ, depending on service language, platform, etc. In the Grid service environment, the Grid Service Administrator (GSA) handles these issues who is a "middleman" for service composition and provision.

Service design and deployment: Web services are designed using a certain development tool, language and platform, usually restricting client access to similar environments. The GSOA is a language and platform independent concept that allows composition of services from different environments, although the current reality of this is more tightly coupled with the most recent version of Globus.

There are additional features as well, but these are some of the most pertinent and worthy of analysis. For further clarification, note Figure 10 showing a theoretical client interaction with different, existing Web services vs. that of a client accessing different Web services via a Grid computing environment: A client wishing to connect to different Web services, after locating them via a Web Service Directory (UDDI, for example), must then connect to each service separately. In contrast, a series of separate Web services deployed in a Grid environment can be accessed by that client through one connection with the Grid Service Hosting Server. Clearly, the Grid Service Oriented Architecture provides a more flexible, less-complex client experience.

By illustrating how much more stable and preferable the GSOA is for client connectivity, access, security, and other related B2B transactions, it is proven that the architecture used in this project shows the future and viability of this type of SCM system development as a typical example of Business-to-Business application integration.

5 CONCLUSION

As evidenced by the results discussed above, the two key questions put forth have been successfully answered. Firstly, an OGSA can be applied to developing a SCM system as illustrated by the example implementation previously discussed. Although systems of this nature have been implemented before in a non-Grid environment, this project shows a concrete example of a previously theoretical implementation in a real world application. The resulting system, through implementation and testing, further goes to argue the idea that this represents the future of Web services, and a more stable environment for B2B application integration.

Secondly, the viability has been shown regarding the OGSA as the future in developing a typical example of B2B application integration. The detailed examples of the differences between the WSOA and GSOA concepts, coupled with the benefits of the singular administrative environment of the OGSA has shown this to be a more stable, desirable environment for B2B system development, integration and deployment. In addition, as future systems are designed with this architecture, their use will become more commonplace.

The result of this project show OGSA and SCM as a viable, working example as to the future of SOC, but is intended as an initial foray into the studied concepts upon which to conduct further research and experimentation. Continued work building upon the complexity of this system in concert with advances in OGSA, Web service framework standards, and more widespread use of their concepts will greatly enhance the advancement of the principles put forth by this research.

REFERENCES

Alonso, G. Casati, G. F., Kuno, H., and Machiraju, V. Web Services: Concepts, Architectures and Applications, Springer Verlag, 2004, pp. 315-319.

Blythe, J., Deelman, E., Gil, Y., Kesselman, C., Agarwal, A., Mehta, G., and Vahi, K. The Role of Planning in Grid Computing, American Association for Artificial Intelligence, 2002, pp. 1-10.

Foster, I., Kesselman, C., Nick, J. M., and Tuecke, S. Grid Services for Distributed System Integration, Computer Magazine, June 2002, pp. 37-46.

Foster, I., Kesselman, C., and Tuecke, S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations, International Journal of High Performance Computing Applications, vol. 15, no. 3, 2001, pp. 200-222.

Foster, I., Kesselman, C., Nick, J. M., and Tuecke, S. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Globus Research Website, June 2002, pp. 1-31.

Fox, G. Grid Computing Environments, Computing in Science & Engineering, Volume: 5, Issue: 2, March-April 2003, pp. 68 –72.

Gannon, D., Fox, G., Pierce, M., Plale, B., von Laszewski, G., Severance, C., Hardin, J., Alameda, J., Thomas, M, and Boisseau, J. Portal Architecture: A Portlet Approach to Grid Services, Global Grid Forum, 2003, pp. 2-15.

Schneider, M. Grid Support Tool Install for Windows 2000 Platform. June 2003.

Staab, S., van der Aalst, W., Benjamins, V. R., Sheth, A., Miller, J. A., Bussler, C., Maedche, A., Fensel, D., and Gannon, D. Web Services: Been There, Done That? Intelligent Systems, IEEE, vol. 18, no. 1, 2003, pp. 72-85.