

Design and Development of a Virtual Organization using Globus Toolkit 3.0

Bina Ramamurthy

CSE4/587 Information Structures

Due Date: 4/18/2004 by mid-night.

Purpose:

1. Understand the components and functions defined by Open Grid Services Architecture (OGSA).
2. Get hands-on experience, working with an implementation of OGSA in Globus Toolkit 3.0 (GT3).
3. Understand the concepts of virtual organization (VO), service definition and service oriented architectures (SOA).
4. Design and implement a grid service for IRS tax filing (as discussed in project 1).
5. Write a Java application to test the service developed in step 4.
6. (optional) Enhance the features of the service by adding logging, notification, security and other persistence services offered by grid framework.

Preparation:

1. Download GT3 and install it project space. Work with the samples in the download. You should be able to run grid services in the samples directory by starting the GUI browser for Globus services.
2. Understand the technology underlying Globus: its architecture and application models.
3. Download the GT3 tutorial that explains how to write a real grid service.
4. All these can be done in your project space.
5. You are also given accounts on LinuxGlobusGrid put together by KenSmith at CSE department. Make sure you have accounts on this grid by logging into “cerf”, “mills” or “vixen” from host machine. You will “ssh” into these machines.

Technology details:

Open Grid Services Architecture ([OGSA](#)) defines the components of a grid service and Open Grid Services Infrastructure (OGSI) specifies the functionality. Globus Toolkit 3.0.2 is an implementation of the [OGSI](#). A virtual organization (VO) supports one or more grid services by sharing resources from various organizations.

A grid service is a web service with features as shown in Figure 1. Basic service is enhanced by standard functionality specified by OGSA. In other words, a grid service can provide in a standard way logging, notification, service data, routability, security etc. These standard functionalities enable the seamless interaction of grid services in a global large scale and high density distributed system. Basic application model is also enhanced by collaborative models, and competitive models with such higher level capabilities as negotiation and mediations. These are initial steps towards commoditization of services and their availability as transparent utilities similar to electricity and water utilities. Such a model will certainly impact the society in a very significant way. Benefits of computers will be experienced by masses without any need to explicitly learn about computers or computing.

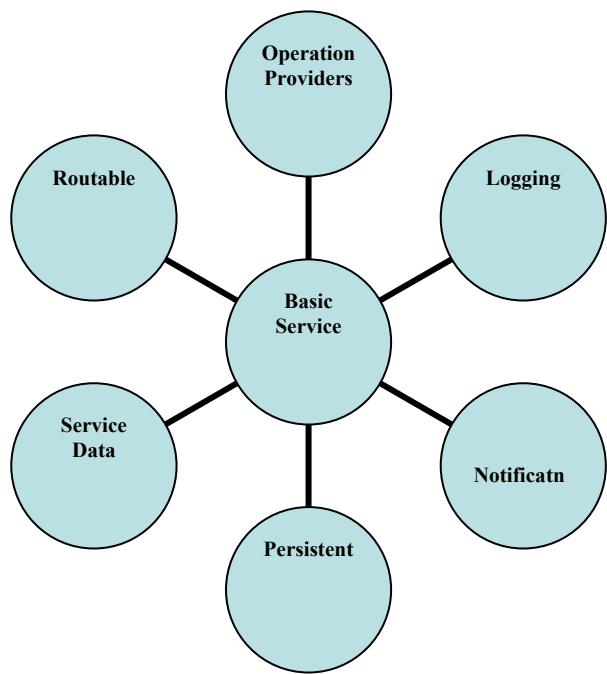


Figure 1 Features of a Grid Service

Assignment (What to do?):

You will implement virtual organization that will feature tax return filing service. Logical specification of the service is the same as given in Project 1.

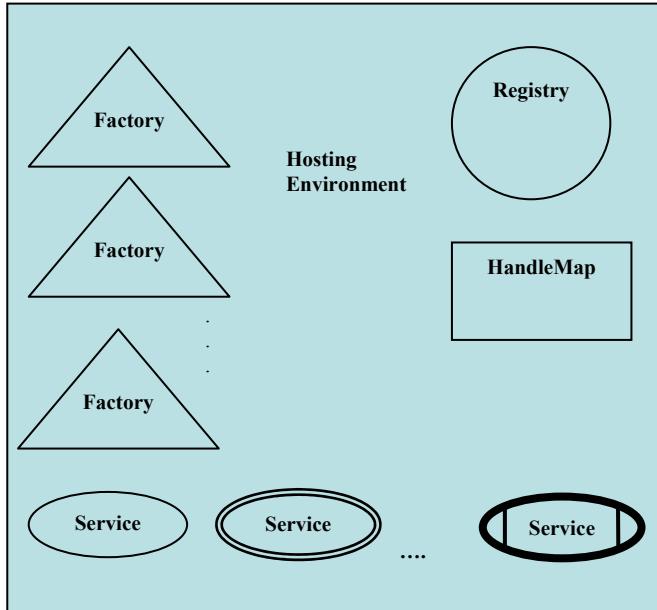


Figure 2: Virtual Organization with Distinct Symbols for Components

You will implement VOs for IRS, Employer, Bank, and Personal profile. IRS VO supports a grid service to perform the tax returns, and each of the other VOs support grid services to perform their respective operations. Let the IRS grid service be a logging grid service and bank service be adorned with notification feature. That is, bank will be a notification sink for messages from IRS. Other than that it is optional for you to add other OGSI features. Figure 2 shows a generic VO with newly defined symbols (by Bina Ramamurthy) for the various components: (service) Factory, Registry, HandleMap, three types of services (simple, complex, and end-to-end service) and the hosting environment. Figure 3 shows the IRS VO composed out of many such VOs shown in Figure 2.

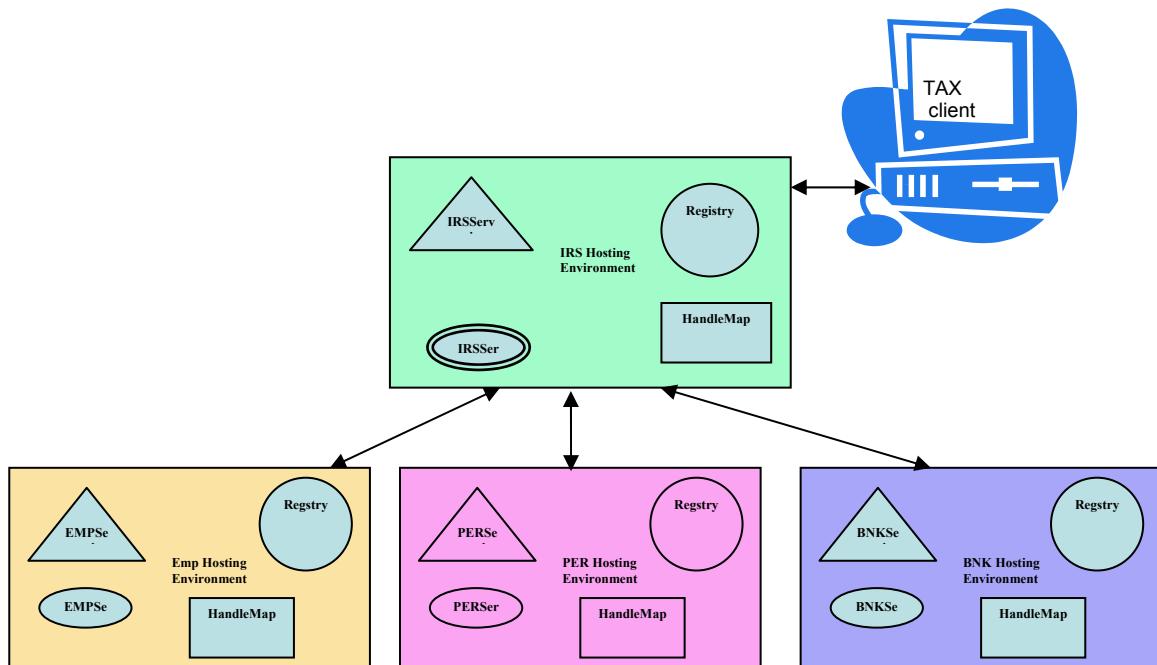


Figure 3: Tax Filing System Architecture

Use Scenario:

Any client who wants to file tax returns uses his/her communication device (a computer, cell phone, pager, telephone, PDA etc.) and authenticates himself/herself by sending appropriate information such as social security number. Then he/she authorizes filing of returns. Tax client then acts as a proxy for the user discovers and instantiates complex service IRSService which in turn invokes the EMPService, PERService and BNKService to accomplish the tax return filing.

Your assignment consists of these parts: You will implement the bottom three grid services independently in your project space and test them. Then write IRS Service that uses three services and accomplishes tax filing. You will work in groups of not more than two people.

Analysis and Design:

Server side: Research and analyze the problem to understand the requirements. Represent the system requirements using UML (Unified Modeling Language) diagram. For each of the VOs identify the entities, processes and rules. Discover classes needed to implement the processes and entities. The rules are typically represented by methods in the classes. Represent the classes and relationship among them using a UML class diagram. Implement very simple grid services with simple files for persistence.

Client side: Write a simple command line client. You may discovery services to locate your IRS service.

Implementation steps and details:

- 1. Getting used to building grid services:** Work with Globus tutorial and understand building grid services. You may use directory structure used by the tutorial or Globus core.
- 2. Building systems using build tools such as Ant:** In order to tackle complexities in configuration and deploying server-side applications, you will need to use special build tools. [Apache Ant](#) is a XML-based build tool which is similar to “make” utility that most of you are familiar with. This topic will be covered during the recitation this week. Work on simple files to familiarize yourself with the Ant build tool.
- 3. Study and understand grid services building and deployment.**
- 4. Design, implement and test your Virtual Organizations and test them.**
- 5. Deploy the integrated system:** The various components listed above were deployed and tested individually. Your final application will use VOs implemented by other groups. So we will need well defined interfaces. Test the individual modules before assembling into a VO application.
- 6. Create .gar (grid archive) for each VO.** Please follow strict naming conventions: usernamePerService.gar, usernameEmpService.gar, usernameBNKService.gar, usernameIRSService.gar. Name other files based on this naming convention.
- 7. Work in Groups:** You will collaborate in groups to implement a VO tax return filing. You can assume any interface and helpers required.
- 8. Practice good programming style:** Finally, practice all the good programming styles that you learned in the lower-level courses.

Submission Details: Use the electronic submission program that corresponds to your class (cse4/587). Submit all gar files.

submit_cse587 xyz.gar at the Unix prompt.

Documentation and Report: See [report details](#).