Design and Development of a Federated Information System

Bina Ramamurthy

CSE4/587 Information Structures **Due Date: 2/24/2004 by mid-night.**

Purpose:

1. Design and develop a multi-tier distributed system offering remotely accessible services.
2. Understand the components, the core technologies, the architecture and the protocols enabling a J2 Enterprise Edition (J2EE)-based distributed system.
3. Design and implement system processes using Enterprise Java Beans (EJB).
4. Understand the process of preparing and deploying an interoperable remote service.
5. Build a Federated Information SysTem (FIST) through interoperation of several autonomous distributed systems.

Preparation:

1. Get a clear understanding of multi-tier distributed systems. (See lecture notes).
2. Understand the technology underlying the J2EE: its architecture and application models. See http://java.sun.com/developer/onlineTraining/J2EE/Intro2/j2ee.html
3. Learn how to use the XML-based build tool Ant at http://ant.apache.org/
4. Understand the role of deployment descriptors. The deployment descriptors are XML files used to configure runtime properties of an application thus relieveing application to deal only with the programmatic details.
5. Learn to use the application interface to the Oracle database using embedded SQL and JDBC. Alternatively you may use a file-based simple database Cloudscape.
6. Download and install Macromedia JRun4 Devloper edition and then the Updater2 (service pack). JRun4 is a J2EE compliant software environment for developing distributed systems. This can be done either or both in the project space that will be allocated to you and at your home, if you have the facility.

Technology details:

J2EE offers a suite of software specification to design, develop, assemble and deploy enterprise applications. It provides a distributed, component-based, loosely coupled, reliable and secure, platform independent and responsive application environment. It encompasses technology solutions for all tiers of a distributed system: client tier, web tier, (business) logic tier, and enterprise information system (database) tier. Sun Microsystems Inc. provides a reference implementation of J2EE compliant environment and many businesses offer fine products such a Macromedia JRun4 and BEA Weblogic for J2EE-based development. For this project, we suggest you use JSP (Java Server Pages) for the web-tier, EJB (Enterprise Java Bean) for the logic tier, and any relational data base (Cloudscape or Oracle) for the data-tier. An *enterprise bean* is a server-side component that contains the business logic of an application. At runtime, the application clients execute the business logic by invoking the enterprise bean's methods. Enterprise Java Bean architecture frees the application developer from having to deal with the system level aspects of an application. Developer can deal with the programmatic aspects of the application while the systemic needs of the application such as data base driver and message queue can be specified declaratively. Ultimate goal of introducing J2EE at this point is to encourage the students to compare it to the grid technology that will be discussed later in the semester.

Assignment (What to do?):

Consider a very common service sought by many people at this time of the year, the tax return filing. It is a yearly duty that many of us love to hate. If we can bring together the organizations that are involved in this tax filing process and allow interactions among them to perform the tax return filing in a trustworthy manner, it will be a great benefit to the society. Assuming that each organization can be modeled as a distributed information system, the above paradigm will allow free and secure exchange of information among the organizations, thus resulting in a Federated Information SysTem or FIST. We will consider four hypothetical organizations as shown in Figure 1: (i) Personal profile system, (ii) Employee information system, (iii) Banking information system and (iv) Internal Revenue System (IRS). We refer to an organizational system as a Virtual Organization (VO) following the terminology grid technology uses.

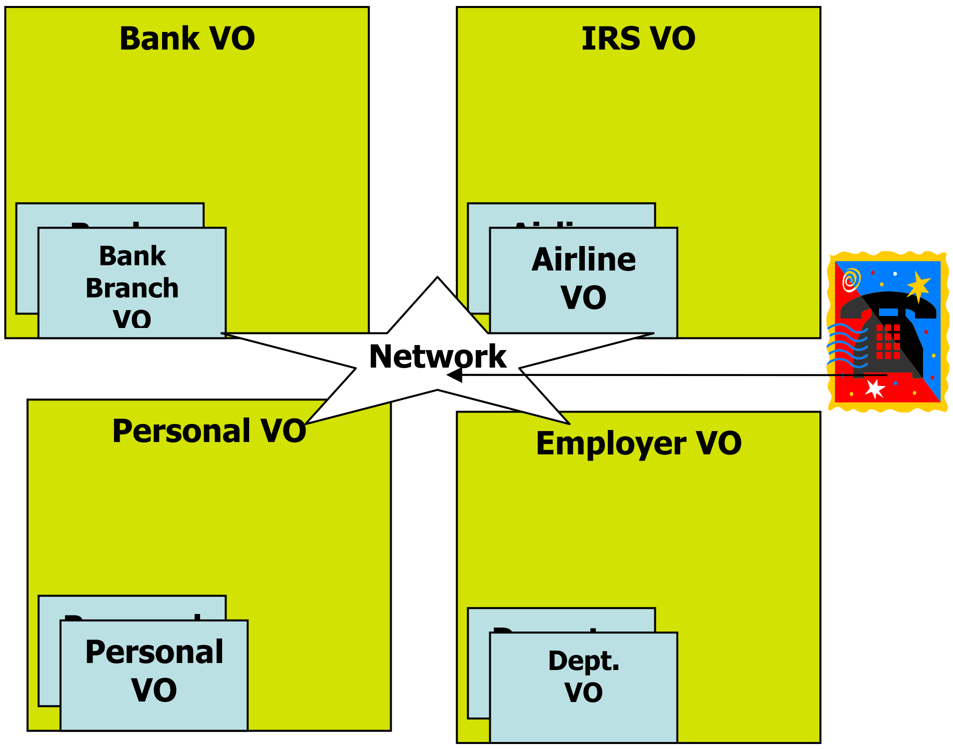

Figure 1: Application Model of a FIST

A person who wants file a tax-return calls up a number and authenticates himself/herself with appropriate personal information (say, last five digits of social security number and mother's maiden name) and authorizes a proxy to file his/her tax return and collect all the necessary information from the FIST shown in Figure 1. Typically there may not be any more interaction needed from the user. Any information needed by the tax filing process is automatically gathered from the organizations collaborating in the FIST. User interface can be any device accessible to a user; however you will use a simulated interface. Determine the user profile using the authentication provided by the user without any explicit request. Gather user information based on this profile. Make decisions and selections to come up with best solution based on user profile and the data collected. Determine payment methods based on the user information and complete the transactions. Notify user if notification were requested. Log the status of the process and any anomalies.

Your assignment consists of two parts: (i) design and implement one of the VOs in the Figure 1 and (ii) write a FIST application that implements tax return filing. You will work in groups of not more than two people. Implementation of the individual VO will be completed by an earlier due date of 2/18/2004. You should submit a J2EE-based tested and operational VO by this date. Then each of the group will work their own FIST application that provides a user interface and interoperates among the VOs to provide the service of filing tax return. This will be submitted on the posted due date of 2/24/2004.

Analysis and Design:

Server side: Research and analyze the problem to understand the requirements. Represent the system requirements using UML (Unified Modeling Language) diagram. Choose one of the systems (VO) for your further design. 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. Decide which among the classes will have methods that will be exposed to the users. Typically these will be implemented as enterprise components (EJBs in our case). Design a relational database to store persistent entities. The design document at the end of this phase will have use case diagram(s), class diagram(s), and a diagram (Entity-Relationship diagram) representing the database design. These documents have enough information to start coding.

Client side: Design a simple interface with a client-tier and web-tier combined for the VOs (Reminder: Each group will design only one VO). However, design a creative user interface for the FIST (overall system).

Implementation steps and details:

1. **Getting used to building client-server systems**: When you implement a simple cient side application program there are just two steps involved: compile and execute the code. In a client-server system, you will have to take care the server side as well as the client side. On the server side, you will compile the code, generate stubs or proxies using special compilers, deploy the service, register and publicize the service for the clients to use. On the client side you will prepare the client code with appropriate stubs, and during excution lookup the service needed and use it. You will notice that besides simple compile and execute, configuration and deployment of a service are important issues to be reckoned with.

2. **Working with the relational database and embedded SQL:** In this project you will store the data in a relational table and access it using SQL statements embedded in Java lanaguage. Work on a simple java program to refresh your knowledge about accessing the Oracle database. See http://www.cse.buffalo.edu/local/Consulting/Orcale.html for examples and access details.

3. **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 similar to "make" utility that most of you are familiar with. This topic will be covered during the recitation this week. Work on simple simple files to familiiarize yourself with the Ant build tool.

4. **Study and understand Enterprise Java Bean building and deployment details:**

   a. You will user Macromedia JRun developer edition for the J2EE components. Download details will be discussed during the recitations. They are quite simple. You can work at home by downloading one into your personal computer and bring the deployable units to school for deployment.
   b. Study the examples in the documentation that comes with the JRun installation.
   c. For the database you may use the database that comes with JRun or Oracle database.
   d. For the client-tier we suggest that you use JSP. We will cover JSP and servlet during the recitation.
   e. While you have choice of technology in implementing data tier and client-tier, it is required that main exposed business logic should be implemented using EJBs. However, utilities supporting the business logic can be implemented using regular Java classes.
   f. It is very important that you understand the concept of remote method call, name resolution, registering and lookup. The concept of component programming using EJBs is also equally important. We will discuss these with examples during lecture.

5. **Design, implement and test your Virtual Organization:** Using the frame work given in the Step 4 above design the VO of your choice. This is expected to be the most time consuming part of the project due to the novelty of the topic.

6. **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.

   a. Test the individual modules before assembling into a VO application.
   b. The final application should single-click accessible from the web.

7. **Work in Groups:** You will collaborate in groups to implement a FIST for tax return filing. The protocol for interaction within and among groups will be clearly specified.

8. **Practice good programming style**: Finally, practice all the good programming styles that you learned in the lower-level courses.

Submission Details:

Create a project1 directory and use that as the working space. Let the code directory tree be located in this directory. Let the design be represented by an integrated class diagram and presented in a file project1.pdf. Provide internal documentation using javadoc style comments. You will create a README file containing the details of the package and processing. Zip the project1 directory and submit the resulting zip file, project1.zip. Making sure that you current directory contains your project1 directory, you can create this file as follows:

```
zip -r project1.zip project1
```

Use the electronic submission program that corresponds to your class (cse4/587). For instance students in cse587 will submit by typing

```
submit_cse587 proejct1.zip
```

at the Unix prompt.

Documentation and Report: See report details.