Due: 2/27/2011 by midnight

PROJECT 1: DESIGNING AND DEPLOYING A SERVICE-BASED DISTRIBUTED SYSTEMS

1. Purpose:

1. To understand the fundamentals of a distributed systems and remote procedure call.
2. To understand the components, core technologies, architecture and protocols that enable a Web Services-based distributed system.
3. To design and implement a Web Service.
4. To understand the process of preparing and deploying a remote service.
5. To work with a relational database system.

2. Preparation before lab:

1. Read and understand fundamentals of a client/server distributed systems.
2. Understand the foundations of remote procedure call by studying and experimenting with remote method invocation (Java RMI): http://java.sun.com/developer/onlineTraining/rmi/

3. Study the Web Services architecture and associated protocols.
4. Learn the fundamental of a relational database and designing relational tables. We will use MySQL for the database needs of our application. Learn to use the application interface to a relational database using embedded SQL and JDBC.
   a. http://java.sun.com/docs/books/tutorial/jdbc/basics/index.html

5. Choose an IDE (Integrated Development Environment) to work with. You may choose to work with Eclipse or Netbeans.
   a. You may work on your lap top or on project space that will be allocated for you on CSE machines. This project space is accessible from CSE machines.

6. Finally, you must have a clear understanding of a client-server system operation.

3. Web Services Technology:

Web Services technology provides a standard means of building a distributed system over the Internet. In simple terms, it provides a means for a sophisticated remote procedure call. The sophistication arises out of the elegant mechanisms it supports for enabling:

1. Various transparencies (platform, language, and hardware) using open standards,
2. Application to application data exchange and interoperability, and
3. Creation of composite web services from a set of simple web services.

The significant difference between regular HTTP-based technologies and Web Services is the standardization realized through the XML and SOAP (SOAP and XML over HTTP). Recently Representational State Transfer (REST) is another protocol gaining popularity. You may use either one of the protocols for WS invocation. All these advantages make Web Services technology ideally suited for large-scale enterprise level application integration.

Web Services specifications are defined by the World Wide Web Consortium (W3C). Many vendors including Oracle and Microsoft (.net) have frameworks for building and deploying Web Services.

4. Assignment:

Build a multi-tier distributed system comprising two major sub-systems

1. A **simple data acquisition system** and
2. A **Web Services** based **web application** that processes and serves the data collected.

The two sub-systems are *loosely coupled* via a database. The block diagram of the system you will implement is given in Figure 1.

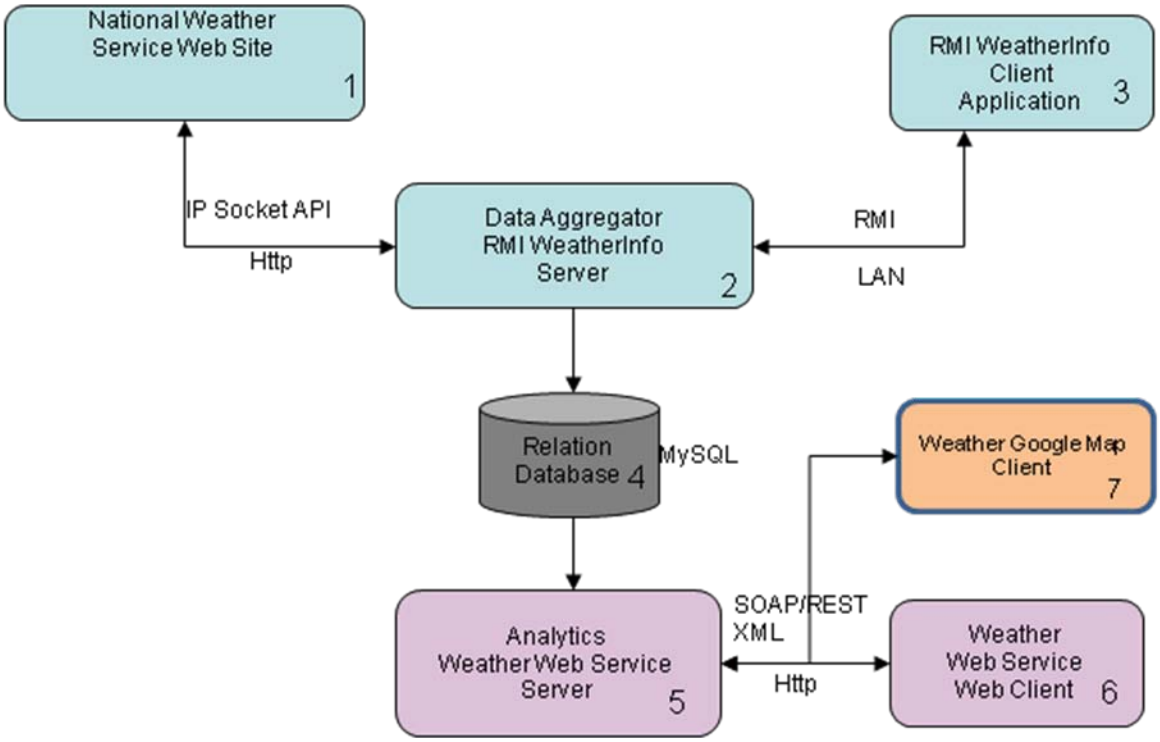

Figure 1: System Model of the Weather Service Distributed System

The national weather bureau updates the weather conditions at various cities on its web site (*box 1 in Figure 1*). You may either use XML feed from weather.com or RSS (XML) feed from weather.gov or any other weather sources.

The Server (*box 2*) streams in the page and parses it for the relevant data and stores it in a persistent storage. Module 2 will parse the XML or RSS feed extract the relevant data and store it in the oracle database. You will have design a suitable relational schema to store the data. You will have collect data for at least 7 days, before you can interpolate the data for future. So module 1 and 2 are time-critical. I expect you to complete these two modules by 2/6/2011. RMI client in the box 3 can be a simple application client accessing the data aggregated.

The persistent storage (box 4): You are required to update the code to accumulate the data for a period of over **at least 1 week** (or *any 7 days*). You may design the server to be a persistent one that automatically collects the data once every day and stores the data in a

database. The data collected should be stored in a **relational database** (*box 4*) on *MySQL*. The daily data collected can be transformed into visually appealing graphics similar to the demo shown in class.

Analytics (box 5): In the **Web Services** part of the system, the data collected in the database will be processed by the server (*box 5*) for such information as average temperature for a given city, and the temperature on a particular day for a city. The city can be specified by either the text name or zip code.

The simple Web Services client (*box 6*): will be able to query the server for various information related to the data collected. The information requested could range from average temperature, high, low and condition for a give city. Your task is to design and implement the complete Web Services-based system indicated by *boxes 4, 5 and 6* of *Figure 1*, and test the operation of the integrated system depicted in Figure 1.

Extend the project given above: An additional and important component for the project grade is the box 7. In this part, you are required to use the weather information and other (one or more) web services available on the net to generate useful intelligence. An example of this given below:

1. A user inputs "From" and "To" on, you will have to figure out the weather predicted for a given route and display it in a user-friendly form. You may use a Google Map API and other web services available online.
2. You can also interpolate the weather forecast from the data collected for the previous week and some reasonable weather prediction model.

5. Project Implementation Details and Steps:

1. **Learn the foundational concepts in building *client-server* systems**:
   - When you implement a simple *client 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 of 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 execution lookup the service needed and use it.

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 the Java programming language (using JDBC). Work on a simple java program to refresh your knowledge about accessing the Oracle/other relational database.

3. **Study and understand the Web Services building and deployment details:**
   - Most IDEs come with servers to deploy the web services. **Please try the tutorials on Web services on the IDE you have chosen before you start working on the project.** If you are using Linux you will have to set up Tomcat server in your space.

4. **Design, implement and test your weather Web Service:**
   - Using the example given in the Step 4 above design the Web Service for dispensing and answering user queries about the weather information of various cities. This is expected to be the most time consuming part of the project due to the novelty of the topic.

5. **Deploy the integrated system:**
   - The various components listed above should have been deployed and tested individually. In this step you will run the entire integrated system. The weather server part can be scheduled to acquire data once a day to update the database that will be used by the Web Service part.

6. Project Deliverables:

1. A Weather **Server** that periodically schedules and updates a relational database with weather information for a set of cities.

2. A simple **RMI Client** that connects to the Server and obtain, and display latest weather information for a set of cities.

3. A **Web Service** that is able to query the relational database for different information. Requirements include (but need not be limited to):
   a. Weather for a particular city on a particular day
   b. Average weather for a particular city over a given period
   c. Ability to convert between English and Metric weather units
   d. **High/Low and condition** for a given city.

4. A **Web** based **client** created using JSP pages or any other visually appealing graphics that allows a user to view and use the various services offered by your web service.
5. The intelligence Client that provides sophisticated analytics and service such as weather along a given route.

7. Submission Details:

Create a compressed deployable distribution of your project. Submit it online.
You should include all the details about the technology requirements to deploy and use your module and also a readme file providing clear instructions as to how to deploy and use your software.

More details will be provided as the deadline near. There is a lot of new stuff you will have to pick up. You will definitely need the whole month.

8. Miscellaneous

- Start working on modules in an incremental fashion.
- The modules in figure1 are color-coded for this purpose.
- Attend the recitation and ask questions.
- You may discuss the problem with fellow students, however NO code sharing is allowed. You cannot email, you cannot file transfer and you cannot text or tweet.
- You may have to research on the material needed for the project, do not expect it to be given to you!
- Develop best design and programming practices.
- Submit the parts of the project as you complete them. Don't wait till the last minute.
- **DUE Date: 2/27/2011**