

Purpose:

1. To understand the components, the core technologies, the architecture and the protocols enabling a Web Services-based distributed system.
2. To design and implement a Web Service.
3. To understand the elaborate and complex process of preparing and deploying a remote service.

Preparation before lab:

1. Read and study the Web Services architecture and the associated protocols:

<http://www.w3.org/2002/ws/>

There is another article available as a first chapter of the Web Services tutorial offered by Sun Microsystems at <http://java.sun.com/webservices/tutorial.html>

2. Learn how to use the XML-based build tool Ant at <http://ant.apache.org/>

3. Understand the role of deployment descriptors. The deployment descriptors are XML files used to configure runtime properties of an application thus relieving application to deal only with the programmatic details. A simple reading on deployment descriptors can be found at

http://www.systinet.com/doc/wasp_jserver/waspj/deployment_descriptors.htm

4. Learn to use the application interface to the Oracle database using embedded SQL and JDBC.

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

Web Services Technology:

Web Services technology provides a standard means ([SOAP](#), XML over HTTP) 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 (i) various transparencies (platform, language, and hardware) (ii) application to application data exchange and interoperability, and (iii) compositability of complex web services from a set of simple web services.. The significant difference between the regular HTTP-based technologies and Web Services is the standardization realized through the XML and SOAP. Web Services Definition Language (WSDL) is an important standard supported that allows for standard definition of services. All these make Web Services technology ideally suited for large-scale enterprise level application integration.

Web Services specification is defined by World Wide Web (W3) consortium in terms of (i) Web Services architecture requirements, (ii) Web Services architecture, (iii) Web Services glossary, and (iv) Web Services architecture usage services. Many vendors including Sun Microsystems (Sun One) and Microsoft (.net) have frameworks for building and deploying Web Services.

Assignment:

Build a multi-tier distributed system comprising two major sub-systems (i) an RMI and simple data acquisition system and (ii) a Web Services based web application processing and serving 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. The RMI part of the project is adapted from the Weather service problem described in the fourth edition of *Java: How to Program?* By Deitel and Deitel.

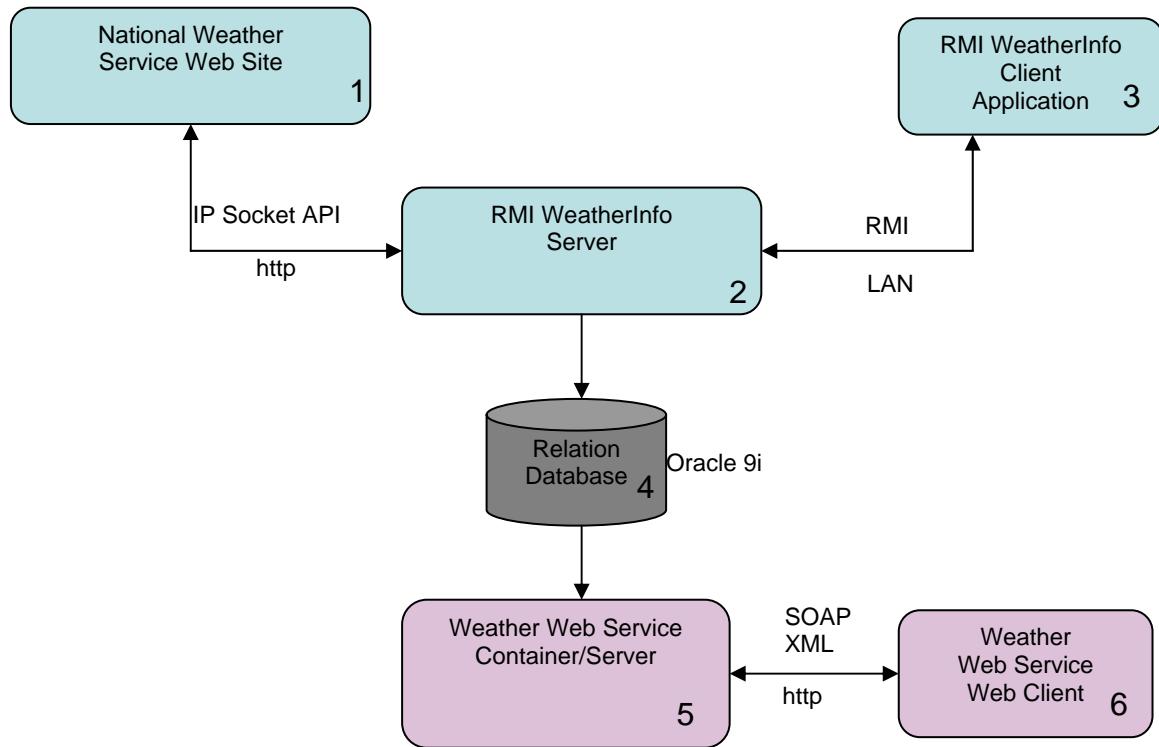


Figure 1: System Architecture of the Weather Service

The national weather bureau updates the weather conditions at various cities once every day on its web site (box 1 in Figure 1) at <http://iwin.nws.noaa.gov/iwin/us/traveler.html>. The RMI Server (box 2) streams in the page and parses it for the relevant data and stores it in a persistent storage. (The details of the RMI and existing code base and a simple frame work were discussed during the lecture and are available at <http://www.cse.buffalo.edu/gridforce/courses.htm>) The persistent storage in the sample code is a simple file and the data stored is just the weather data for one day. You are required to update the code to accumulate the data for a period of over at least 1 week (or

any 7 days). The data collected will be stored in a relational data base (box 4) on Oracle 9i. The daily data in the sample code is served to an RMI client (box 3) which simply renders the ASCII data provided by the weather bureau in the form of visually appealing graphics.

In the Web Services part of the system in Figure 1, the data collected in the data base will be processed by the server (box 5) for such information as average temperature for a given city, and the temperature for a particular date for a city. The Web Services client (box 6) will be able to query the server for various information related to the data collected. Your task is to design and implement the complete Web Services-based system indicated by boxes 4, 5 and 6 of Figure 1.

Project Implementation Details and Steps:

- 1. Getting used to 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. To understand the process study the RMI-based system code and implementation. Deploy it and make sure it works and you understand the various operations. 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 language. 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 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.
- 4. Study and understand the Web Services building and deployment details:** For the Web Services part we will use the Sun Microsystems implementation of the Web Services specification. We have a version of the HelloWorld (in Web Services) available at /projects/bina/cse486. This will provide framework to develop your Web Services client-server system. It has clear directory structure

which you are expected to follow. It has the source code for the server and the client, a build file and a configuration file in XML. Copy the code WSSample.tar into your project space. Unzip it, build the server and deploy the server. Build the client which is a web application and access the service provided by the server. For this step you need to download and install JWSDP 1.2 <http://java.sun.com/webservices/downloads/webservicespack.html>

5. **Design, implement and test your weather Web Service:** Using the frame work 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.
6. **Deploy the integrated system:** The various components listed above were deployed and tested individually. In this step you will run the entire integrated system. The RMI part can be scheduled to acquire data once a day to update the database that will be used by the Web Service part.

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 and also a file that contains the questions and answers for the questions pertaining to the topic of the project that will be given to you later.

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 section. For instance students in section A will submit by typing

```
submit_cse486 proejct1.zip
```

at the Unix prompt.

Due Date: 10/10/2003 by midnight.