Project 1:  
RMI & Web Services  

CSE 486/586  
Sept 28, 2004  

Core functionalities  
- RMI Server must be able to get data from public data source and store into local database. This must be recurring process  
- RMI Client should connect directly to RMI Server and get the most current data  
- Web Service must be able to interact with the database and provide methods that query (and aggregate) data  
- Servlet must be able to remotely make calls to the Web Service and show the data in a more presentable manner

Core functionalities  
- All the above modules must be totally independent of each other and be deployable in separate contexts and still be able to interact  
- All modules should build and deployable / runnable without giving errors

Suggestions  
- Use of ANT is highly recommended. Server addresses and ports must be required to be specified only in the deployment descriptor.  
- Create database in one person’s Oracle account  
- For the RMI Server, you have code that fetches the data once from the public data source. You need to modify this to make it update repeatedly after a pre-defined delay and to add the retrieved data into the database

Suggestions  
- Use java.util.Timer and java.util.TimerTask for implementing this  
- You may need to take care of duplicate data in the database due to the recurring updates  
- The RMI Client can be used almost without any modification (if doing WeatherService)  
- The Web Service can be modeled on top of the example Hello4586Service shown earlier

Suggestions

- Examples of WeatherService methods:
  - getAllWeatherInfo()
  - getAvailableCities()
  - getAvailableForecastDates()
  - getAverageHighLowTemp()
  - getWeatherInfo(date, city)
  - getWeatherInfoByCity(city)
  - getWeatherInfoByDate(date)
- Will post example WSDL snippets

Suggestions

- The Web Service Client can use JSP or Servlets. A few points will be awarded for look and feel
- Soon I will post an example of my code. You can test your server and client against my client and server respectively.