PROJECT 1: CONTENT RETRIEVAL, STORAGE AND ANALYSIS FOUNDATIONS OF DATA-INTENSIVE COMPUTING

Purpose:

1. To understand the components and core technologies related to content retrieval, storage and data-intensive computing analysis as indicated the Fourth Paradigm text [1]
2. To design and implement the solutions for a data-intensive problem using traditional approaches
3. To explore designing and implementing data-intensive computing solutions on a cloud environment: in this case on Google App Engine [2]
4. To review/familiarize students with simple three-tier enterprise application in Java: design, implementation, deployment on an application server such as Apache Tomcat with data stored in a relational database

Problem Statement:

We will consider the countries as listed in the United Nations as the data domain for this problem, though this data may not be high-volume. You are required to create complete application that will: (i) collect a set of relevant data on **all the countries** in the United Nations (UN), (ii) store the content in a suitable form, and (iii) make the bulk data available through web services, and (iv) allow specific queries on the data. You are required to implement (i) simple and (ii) cloud-based solutions for the problem, and analyze and compare the performance of the different implementations.

Preparation before lab:

1. Review your Java language skills by working on the sample application that will be given to you.
2. Learn about web services; how to create them and how to publish data using them.
3. Familiarize yourself with MYSQL as a content storage for the first two non-cloud versions of the application.
4. Finally, you must have a clear understanding of a client-server system operation and also three-tier (web, logic, and database) application development.
5. Get an account on Google App Engine and familiarize with the environment.

Assignment:

You will develop this project in **three phases**:

Phase 1:

Build an application that will consume web services about countries in the world that is available at the United Nations site:
http://www.fao.org/countryprofiles/webservices.asp
It will collect at least these information: country name, capital, country lat/long, GDP, area, land area, population, GDP and Human Development Index. Collected information will be stored in a relational database whose scheme you will design. (Initially we will work with just for the year given in the web services. Later on we may add data for several years as given in the site:http://faostat.fao.org/DesktopDefault.aspx?PageID=550&lang=en)

Phase 2:

(1) Build an information system that will use this data aggregated. The system will publish them as web-services for consumption by web-services clients. These could be simple JSP clients.
(2) Provide some creative and useful functions as web services for the users: rank the countries by GDP, rank them by other parameters, trends, averages. We encourage

you to be as creative as possible. **Analytics could be anything from a simple classification of the data to complex predictive analysis that may require modeling the data.**

Phase 3:

(1) Port the application to the Google app engine (GAE) environment. There are some challenges you have to address when dealing with GAE. The database has to be the BigTable of Google App Engine and not the regular relational DB. There are some restrictions and limits on such characteristics as timeouts for web access etc. You will have to address these issues.

(2) Compare the performance in terms of response time and load handling. That is, scalability of your application has to studied. (We will give you more details later.)

(3) Write a report about the project that contains detailed description of the phases, their design and implementation, and their performance for scalability.

Application Architecture:

The application architecture of the data-intensive analyzer is shown in Figure 1. The modules are numbered 1 -7. The two of the three phases are also shown in that figure. Module 1 represents the Web services (WS) available at United Nation Food and Agricultural Organization (UN FAO). You are free to use other relevant web services, if you wish. However make sure you document it clearly. Module 2 is a web services client that consumes the web services provided by external sites. The data from FAOWs is in XML format and thus have to be parsed (parser in box 3) to extract the data for our application. The data aggregated is stored in a persistent storage (MySQL Database in module 4).

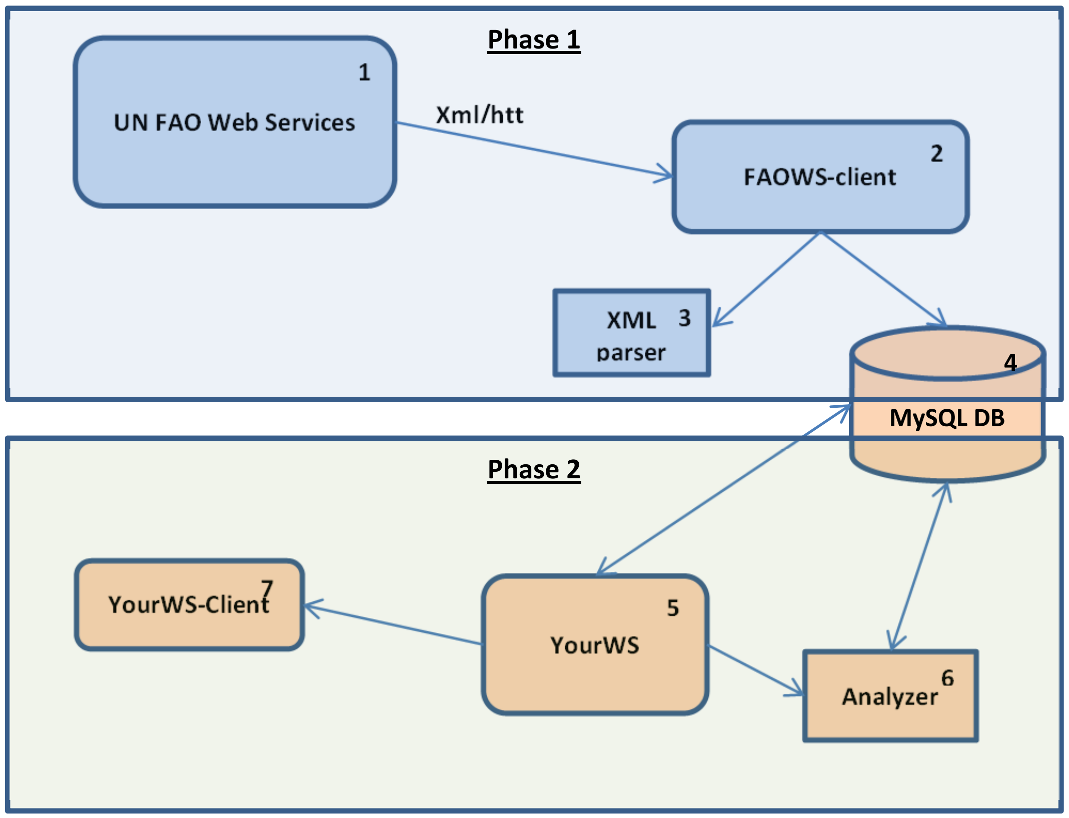

Figure 1: System Architecture for Data-intensive Analyzer

Module 4 provides the loose coupling between the Phase 1 and Phase 2 applications, thus making them independent for design, implementation, testing and administration.

Module 5 is a web services server that processes the data aggregated and makes them available as web service for consumption by clients indicated by Module 7. These clients can be simple JSP clients. For requests from WS-clients that require analysis, for example, “Countries Ranked by GDP”, an analyzer indicated in Module 6 is used. This unit can directly access the database for required data.
In Phase 3, modules 4-6 will be ported to the Google App Engine cloud. MYSQL DB has to be ported to Big Table, and then the Java Application representing modules 5, 6 and 7 can be deployed on the GAE cloud. The Big Table is not relational table and can be accessed (for reading as well as writing) only through API that GAE provides.

Project Implementation Details and Steps:

1. Access the UN FAO site and study the web services it provides. Make a note all the possible data that you acquire from that site. **We allow images and maps also as data.** This will give you some interesting data for analysis and for visualization.
2. Build a simple Java Application to consume one of the web services and prototype module 2.
3. Then parse the information obtained and extract the data. (You can also store the data as XML and work with XML data… we will NOT do that).
4. Decide on the data you want for your database; based on this determine the web services you would consume.
5. Design the database schema for MYSQL database.
6. Now expand the application build in step 2 to obtain the whole set of the data that you need and store them in the MYSQL db. Phase 1 Done. **Due date: 10/1**
7. Write a Java Web Services application that export the data you aggregated to be available as a web service.
8. Delegate any analysis to be done to a utility class Analyzer. In a real world application this analyzer will play a critical role and will house the complex approaches and algorithms used to process the data.
9. Write a simple JSP WS-client that will consume the operations offered by the web service that you designed above. Test it. End of Phase 2. **Due date: 10/8**
10. Phase 3 involves porting Phase 2 artifacts to GAE. **Due 10/15**
11. Documentation including design diagram: **Due date: 10/22**

Project Deliverables:

1. The applications with proper directory structure bundled with the source code, REDAME, and clear instruction to deploy and use the applications. Phase1App.war, Phase2App.war, GoogApp.war (or any other suitable archival format).
2. A report providing all the details of the project. This should have the user’s manual, programmer’s manual and any design diagrams: Prj1Report.pdf

Submission Details:

submit_cse487 files separated by space
submit_cse587 files separated by space

References:

[1] The Fourth Paradigm: Data-Intensive Scientific Discovery**,** Tony Hey (editor), Stewart Tansley (editor) and Kristin Tolle(editor), Microsoft Research (October 16, 2009), ISBN-10: 0982544200.

[2] Google App Engine. http://code.google.com/appengine/, last viewed 9/2011.