

CMSC 424: Application Development Project - Fall 2008

1 Overview

In last couple of years, smartphones (like Blackberry, iPhone) have become quite popular, and are used for email, web browsing, custom applications etc. At the same time, there are many companies that provide the cellular or the Wi-Fi technologies to enable these devices. We focus on the “data usage” for these types of devices. Cellular data transfer is usually done using Edge or 3G. We won’t distinguish between them.

You are to write a web-based application and tools, call it ADP, to manage and broker the connections and the data usage for these devices. Specifically, each smartphone is subscribed to the services from some company (Verizon, ATT etc.). Each cell tower and each Wi-Fi access point is managed by some company (Verizon, ATT, Boingo etc.¹). At any point, a smartphone is connected to one cell tower and possibly one Wi-Fi access point. You are to keep track of this information over a period of time, and also manage the billing charges (more details below).

ADP will also serve as a clearing house for the emails. So all emails will be routed to ADP, and it has to decide whether to “push” them to the smartphones or not (based on whether there is an agreement in place to use the cell tower the subscriber is currently connected to²). The cell towers/access point will also send ADP information about how much data usage a smartphone had while it was connected to the cell tower/access point.

More details on the kind of data to be stored, and queries to be supported are provided below.

You are free to change the details, as long as the main functions described here are supported. In case of drastic changes, you should talk to the TA or me first. The project description is intentionally somewhat vague; you are free to decide what kind of interfaces to support etc.

2 Data

A brief description of the data that needs to be stored in the database follows.

- **Subscribers:** Information about the subscribers (names, addresses), what smartphones they have, which company they are buying services from, what kind of contract they have with the company, when does their billing cycle start etc. The contracts can be of at least two types:
 1. Monthly contracts: E.g. \$40 a month gets you 40MB data usage on Edge/3G, **and** 500MB on Wi-Fi. After that, you are charged \$10 per MB on Edge/3G, and \$1 per MB on Wi-Fi. There would be multiple levels of such contracts (\$20 for the cheapest plan to \$100 which is unlimited data).
 2. Pay-as-you-go: E.g. \$20 per MB on Edge/3G, and \$2 per MB on Wi-Fi.
- **Companies:** Different companies that are providing the services. Some companies only operate cell towers, some only Wi-Fi access points, whereas some do both. Each company has an agreement with some of the other companies about how much to charge for each byte transferred using their technology; the cost will be different for Wi-Fi and Edge/3G (higher for Edge/3G). So if a subscriber is Verizon customer and uses a Sprint cell tower to transfer data, then Sprint will bill Verizon accordingly. Note that Verizon will shield this from the subscriber; the subscriber only has an agreement with Verizon and will pay Verizon accordingly to that rate.

¹See Cell Towers in DC to get an idea.

²The chief reason for that would be that technologies are incompatible... e.g., GSM vs CDMA.

- **Cell Towers/Wi-Fi access points:** Who owns the cell towers (CTs) or the access points (APs), where it is located (latitude/longitude).
- **Connections:** When a subscriber connects to a CT/AP, the CT/AP will send a message to ADP so ADP can make decisions about whether to push emails etc or not. Similarly the CT/AP will inform ADP when the subscriber disconnects, and will also send ADP a message with the total amount of data transferred. The current information about the connections, and also the historical information about all the connections should be maintained in the database. Note that a CT connection may overlap with an AP connection.

3 Tasks & Queries

There are two sets of tasks/queries that ADP needs to support. The main tasks are provided below. You may choose to add and/or substitute other tasks depending on the emphasis of your project.

3.1 Web Interface

The following tasks should be provided using a web interface.

- **Subscriber and Company log-ins:** There will be an account for each subscriber, and also for each company. The accounts must be password protected.
- **Insert/update Companies, Subscribers, Cell Towers, APs etc.:** These web-based interfaces will allow addition/updates of new companies, new agreements between the companies, new subscribers, new cell towers, new APs, etc. New subscribers are inserted by the company (after it logs in).
- **Generate Bills:** Generate a bill for the subscriber over the last month, based on its data usage. Remember that the subscriber only has an agreement with a single service provider, and must be charged according to the rate(s) decided then. See the example contracts above. Ideally, the company rep. should be able to click one button to get billing reports for all subscribers whose billing cycle ended that day.
- **Generate Bills for Companies:** At the end of the month, you have to generate a bill detailing how much a company needs to pay another company, and vice versa. The companies should be able to use this functionality after they log in.
- **Generate Usage Details for a Customer:** A subscriber may log-in and ask to see how much data usage he has had over since the last billing cycle. This information should be provided along with the limits (e.g. “You have used 40% of your Edge/3G limit in just 2 days” etc.).

3.2 Stand-alone Programs

The second set of tasks have to do with managing the messages to and from CT/AP. Due to the volume of the data, these tasks are better done as a stand-alone program that listens on a socket for the messages. We will use a simplified XML format to exchange the messages.

The different types of messages are:

- *CT/AP → ADP: connected:* Sent to you when a subscriber connects to a CT/AP. Such a message could have a simple format like:

```
<connected>
  <subscriber> 1-234-567-8901 </username>
  <time> ... </time>
  <cell-tower> cell_tower_id </cell-tower>
</connected>
```

- *CT/AP → ADP: disconnected:* Sent to you when a subscriber disconnects from a CT/AP.
- *CT/AP → ADP: billing charge:* Information about how much data usage a customer had during a connection. We won't worry about how the CT/AP generates such a message.
- *ADP → CT/AP:* An email. If ADP decides to push an email to the cell tower to be delivered to the smartphone, it has to send an appropriate message to the CT/AP. The message format could be simply:

```
<email>
  <subscriber> 1-234-567-8901 </username>
  <email-content> .... </email-content>
</email>
```

For now, the stand-alone program should simply output (to standard output or to a file) any such message to the CT/AP.

- **Emails to ADP:** An email to a subscriber is essentially routed to ADP (using a similar XML format used above). ADP has to decide whether to push this to the subscriber right now, or whether to wait. For our purpose, the only reason for waiting would be that the subscriber's company does not have an agreement with the CT/AP that the subscriber is currently connected to.

If ADP decides to wait, then the email must be stored in the database, and when the subscriber changes the CT/AP it was connected to, the decision must be re-evaluated.

A single stand-alone program, which talks to the database just like the web server does, should be able to handle all of these tasks (the XML message itself contains enough data to decide what the message is about). For testing purposes, you should also write a small program that sends appropriate message to this stand-alone program.

Populating the tables: You should populate the tables manually with sufficient tuples (10-100 each), and also set up a program for automatically sending updates to ADP, in order to demonstrate the functionality of the system. Feel free to use made-up data. We will supply some initial data for populating the tables.

Extra credit: There are several directions that you can explore to enhance your project for extra credit (upto 20% of the project grade/4% of the total grade). Some suggestions:

- **Deal with Failures:** There could be issues with data delivery with messages being lost etc. Reason about the different types of failures, message losses that may happen in the system, and try to build safeguards around it. For example, you may want to execute a two-phase commit when exchanging subscriber emails.
- **Browse Historical Data:** A service provider would want to analyze the data to decide if there is something it can do to lower its cost of operation. In particular, it would be interested in knowing how much it would save if it installed a cell tower at a specific geographical location. Design interfaces/mechanisms to help the companies make these types of decisions, to renegotiate etc. Use the location information of the cell towers for this purpose (e.g. ADP may suggest that a service provider look into switching from using one cell tower to a nearby cell tower based on the prices they charge).
- **Build a simple App Store:** Analogous to the iPhone app store, ADP could serve as a clearing house for the applications for the smartphones. The app developers will submit their apps and updated versions to the ADP, and the ADP will keep track of what is maintained on what phone, whom to send updates to etc. Also, take a look at Apple's "Ad Hoc" distribution program for iPhones.

Be creative.

4 Rules of the game

- **Groups:** The project is to be done in groups of 2 students. The groups are “self-policing” (e.g., each group is responsible for its own division of labor, scheduling, etc.). *Note: If an unreconcilable problem arises in your group, it is your responsibility to contact both me and the TA as soon as possible.*
- **Assumptions:** In cases where you have questions on the above description, it is acceptable to make assumptions about the application providing that: 1) they are explicitly stated in the report, 2) they don’t terribly conflict with any of the requirements specified above, and 3) they are ”reasonable”. If you have a question about the acceptability of any of your assumptions, check with the TA or the professor.
- **Reports:** A report should be handed in at the end of each phase (Due dates below).
- **Implementation:** The final phase of the project requires a working implementation of the system to be built, tested, and demonstrated. A large part of the project grade depends on the quality of this implementation. The implementation will be done as a client-server system in which a web server runs on your cluster unix account, accepts web queries, and connects to the Oracle DBMS to retrieve the data (you are free to use other tools).

5 Project Phases

Phase	Phase Name	Due Date
<i>0</i>	Group names to the TA	Sept 23, 2008
<i>I</i>	System Analysis and Specification, Conceptual Modeling, Toy Web Application	Oct 23, 2008
<i>II</i>	Implementation, Testing, and Demo	Dec 2, 2008

6 Reports

The Phase I report must contain:

1. Assumptions you have made the enterprise, or any additions/changes made to the specs.
2. The graphical schema using the E-R model, including a list of the attributes for each entity and relationship.
3. The relational schema obtained by converting E/R to relations, and their BCNF or 3NF forms with keys.
4. A description of how you plan to populate the tables (including details of how you plan to parse the XML updates).
5. A toy end-to-end application that uses the database. The sole purpose of this is to familiarize yourself with how to build such an application. E.g. you could populate the database with the Olympics data, and the webpage could provide a text box for typing in a player name (like Google search box), and when the “submit” button is clicked, the results of “select * from Players where name like '%ABC%’” would be displayed (where ABC is the name entered in the text box).

You can either send me and the TA an email with the website address (preferred), or show it to the TA in person. More details will be announced later.

The Phase II report must contain:

1. Phase I report with corrections addressing TA’s feedback.
2. Any revisions made to the relational schema definition from Phase II,
3. A brief summary of your implementation efforts, the tools used etc.
4. The key thing here is the demo of the system in person. Details will be announced later.