Field Session 2012
Newmont Mining Corporation
Bump-O-Meter

Lorenzo Gallegos

Joshua Shaw

Craig Soto II

June 14, 2012

Introduction

Newmont Mining Corporation is a company which primarily produces gold. Today, they have mining operations in nine different countries across the globe. Newmont is devoted to the environmentally friendly discovery and recovery of gold. Some of the company's mining operations include open pit mining. Large quantities of earth are removed from these pits using over-sized trucks. These trucks drive along dirt roads that become washed out and bumpy.

Recognizing the need for safety, our client came up with an idea to create a smart phone application to find and measure bumps. The smart phone should collect data, send it to a server, then display the information on a user-friendly web page. Although the idea for measuring bumpiness of roads came from mining trucks in high-traffic open-pit mines, this prototype application was designed to be used for collecting data on bike paths.

System Design

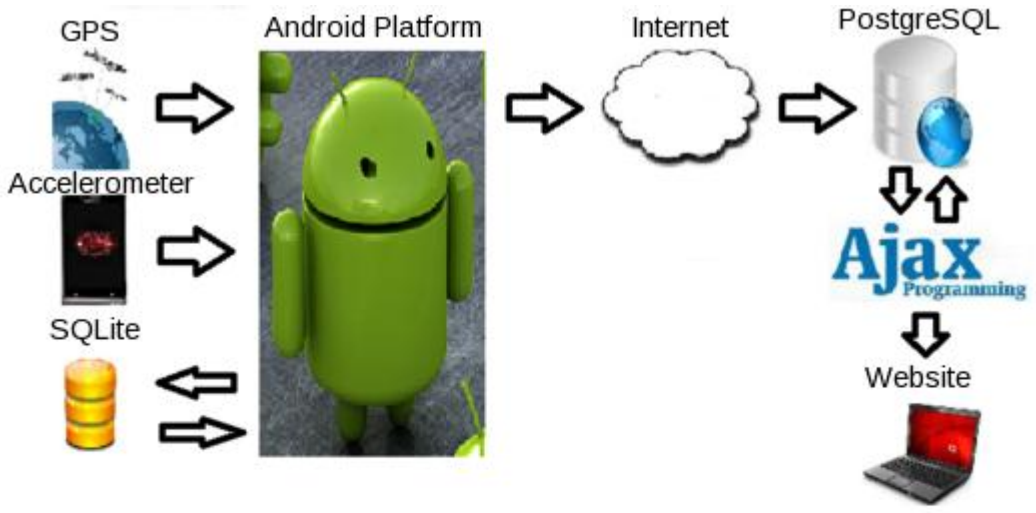
Functional and Non-Functional Requirements

- Application
    - Require the user to input a username and path name
        - Allows the user to find and view their unique data on the website.
    - Record GPS coordinates and Accelerometer data
        - GPS used to plot path on a map on the website
        - Accelerometer used to display bumpiness of path
    - Store data temporarily on the phone
        - This gives the user the option to send data at their convenience
    - Send the data to a central server
        - Server is used to track all data sent from mobile users
        - Website accesses data stored on server
    - Give the user the option to send the data at the end of the bike ride or at a later time
        - The user may not have data service at the end of a ride, the user is given the option to send the data later
    - Display bump feedback to the user
        - Shows the user that the phone is properly recording data
    - Allow the phone to be mounted anywhere during run-time
        - Makes phone position independent of the data collected
    - Present user with basic path statistics upon completion
        - Gives the user some initial statistics about their ride
- Website
    - Allow user to select from drop-down menus
        - Allows the user to narrow their search in a user-friendly manor
    - Display map with path and bumps
        - Allows the user to see their paths and where severe bumps occurred
    - Display chart to convey bumpiness of path
        - Presents user with more detailed bump information
- Non-Functional
    - Website does not have to reload
    - Android application will be written in Java
    - Data will be stored on a PostgreSQL server
    - Android must have a reliable transmission of data to a PostgreSQL server

Software Architecture

Data is initially recorded on an Android device using GPS and the provided accelerometer. Data is temporarily stored locally in a SQLite database. The data is then sent to the PostgreSQL server that was provided by Newmont. Data from the server is then accessed via the web. The data is passed back and forth between client and server to allow the user to view unique path data on the web page. This is all depicted in Figure 1.

Figure 1: Software Architecture



Technical Design

The Bump-O-Meter application stores a significant amount of data during a bike ride in a SQLite database on the phone. Figure 2 shows the SQLite schema for the local database. When a user begins recording path data, the username, date of recording, and trail name are stored in the first table. During recording, the application “listens” for the sensor to change. The accelerometer is sensitive enough to register a change on even the tiniest bump. When the application “hears” a change, it creates a data point with the last known GPS coordinates as well as the acceleration and a timestamp. Each data point is then stored in the second table as they are recorded.

Figure 2: Temporary Phone Database

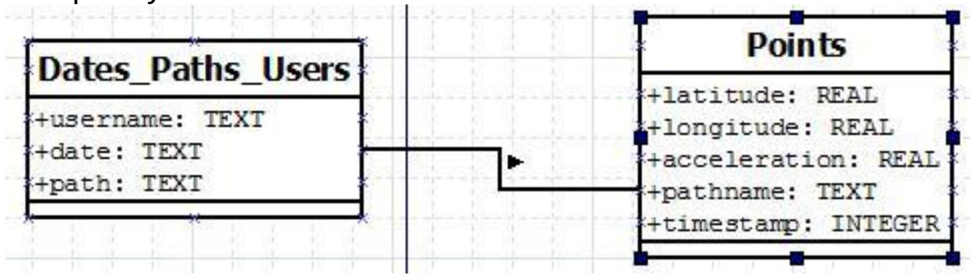
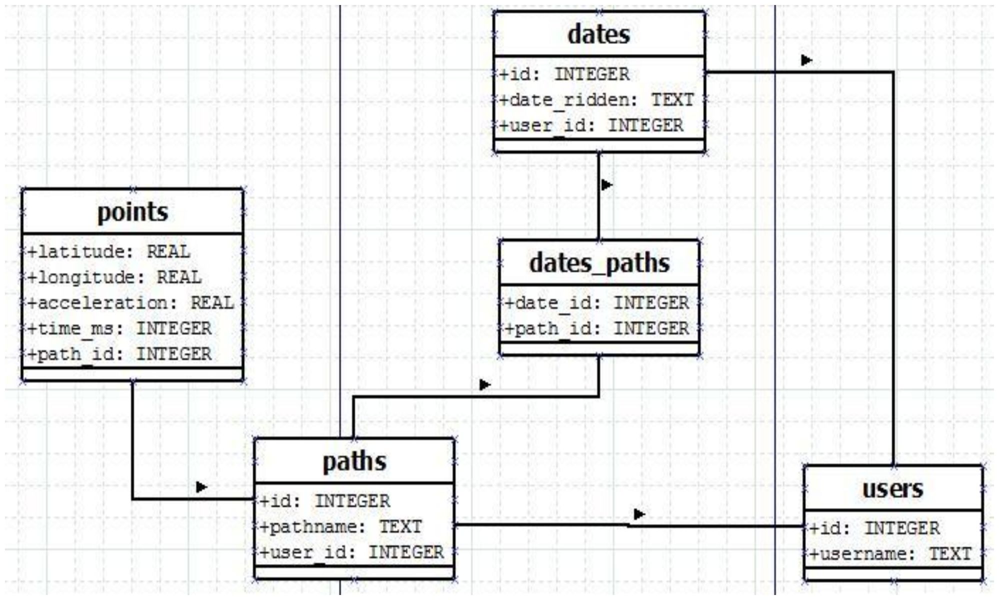


Figure 3 shows the full schema of the database on the PostgreSQL server. The data is stored on the PostgreSQL server in 2nd normal form. The users table stores the name of all users that have uploaded information to the server. The dates table stores each day that a path was ridden. A unique entry is created for each user (ie. Bob and Alice ride a path on the same day, a date entry will be made for each user). The paths table stores the name of each path. A dates_paths join table exists because a user can ride the same path on two different days and a user can ride multiple paths on the same day. Finally, the points table stores all relevant GPS and Accelerometer data associated with a path. See figure 3 for full schema.

Figure 3: PostgreSQL Server



The user first starts the Bump-O-Meter application on his/her Android device and performs a login by entering their username. The user is then redirected to the home page, where they have the option to start recording, send stored data, delete stored data, or log out. If they press the start button the user enters the path name before recording. This is shown in Figure 4.1. Once the user presses the accept button, the app begins recording. General feedback is provided to the user to alert them if they hit a bump. Once the user stops recording, feedback will be provided and they can either choose to send the data now or later. This is shown in Figure 4.2. Whether the user chooses to send now or later, they will be redirected back to the home screen where they are provided with the original start, send data stored, delete stored data, and log out options. If the user chooses to send the data later, and after being redirected to the home screen, chooses to delete the data, the user will be prompted to confirm if he/she would like to delete the data or not. This is shown in Figure 4.3.

Figure 4.1: Login/Main/Path Name

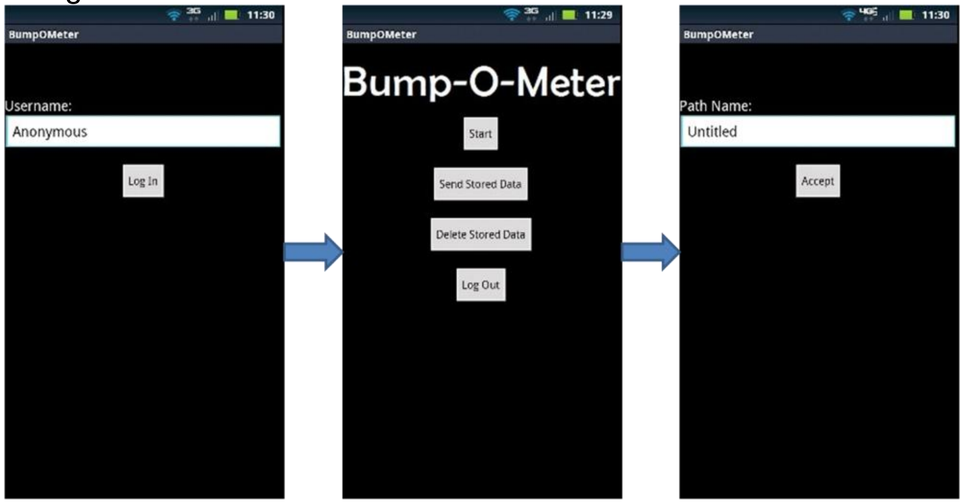
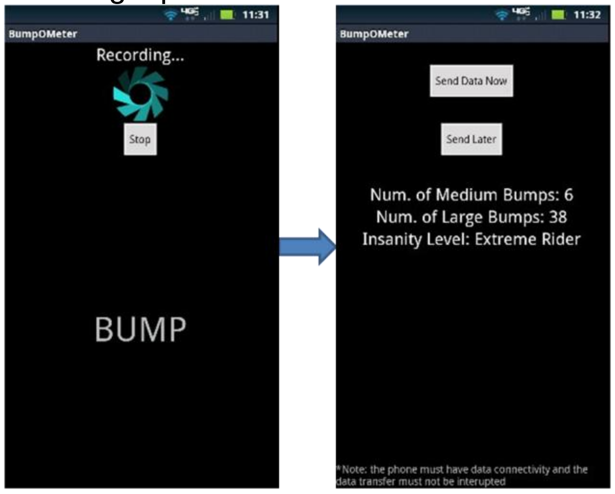


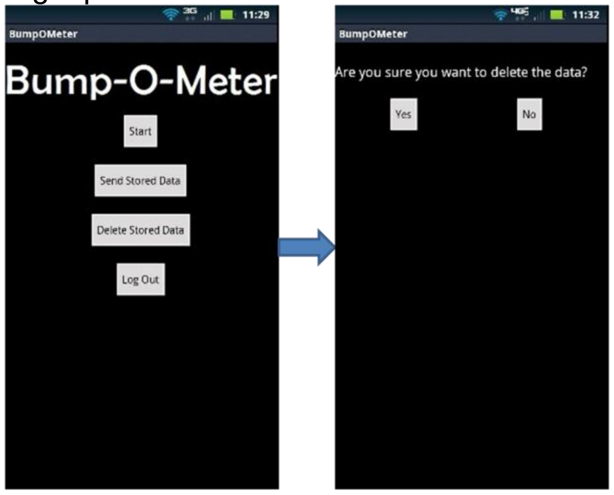
Figure 4.2: Recording/Sending Option



Num. of Medium Bumps: 6
Num. of Large Bumps: 38
Insanity Level: Extreme Rider

*Note: the phone must have data connectivity and the data transfer must not be interupted

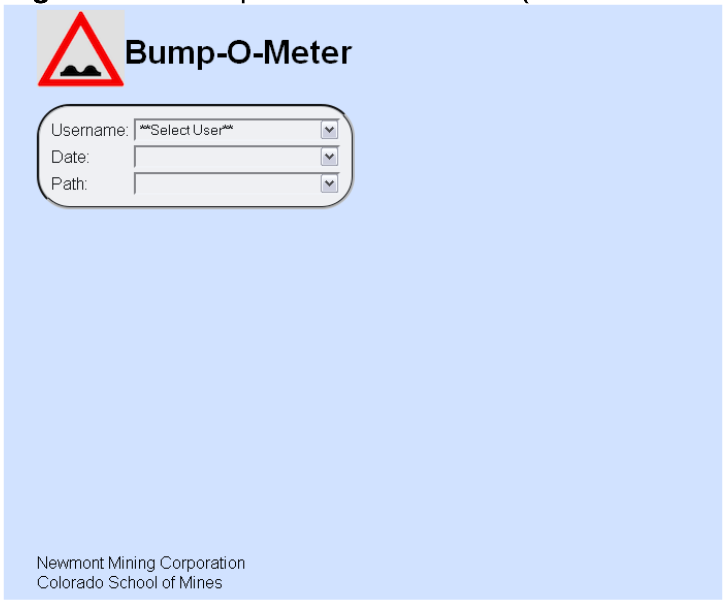
Figure 4.3: Main/Deleting Option



Are you sure you want to delete the data?

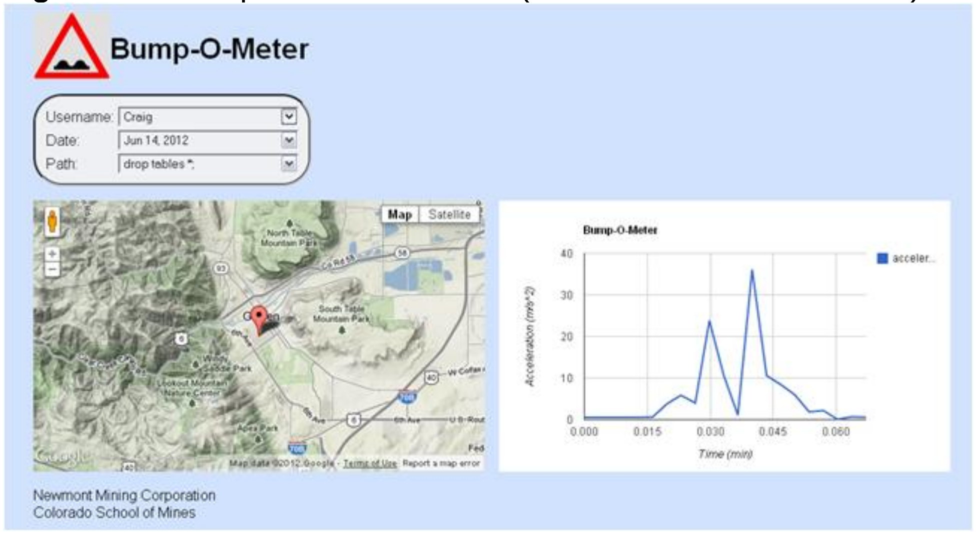
Upon going to the Bump-O-Meter website, the user is presented with a nearly blank page (Figure 5.1), with drop-down menus for username, date, and path. Upon arriving at the web page, only username will have options. Once the username is selected, the date drop-down menu is populated, and by default, the paths drop-down menu is populated for the first date. The user then selects a date and chooses the path name for the path they want to view. Upon selecting a path name, a map and graph will be displayed, conveying the full path information (Figure 5.2). At any time the user can choose to select new user information and the correct data will be displayed. A full flow-chart is provided in Figure 6.

Figure 5.1: Bump-O-Meter Website (before Username is selected)



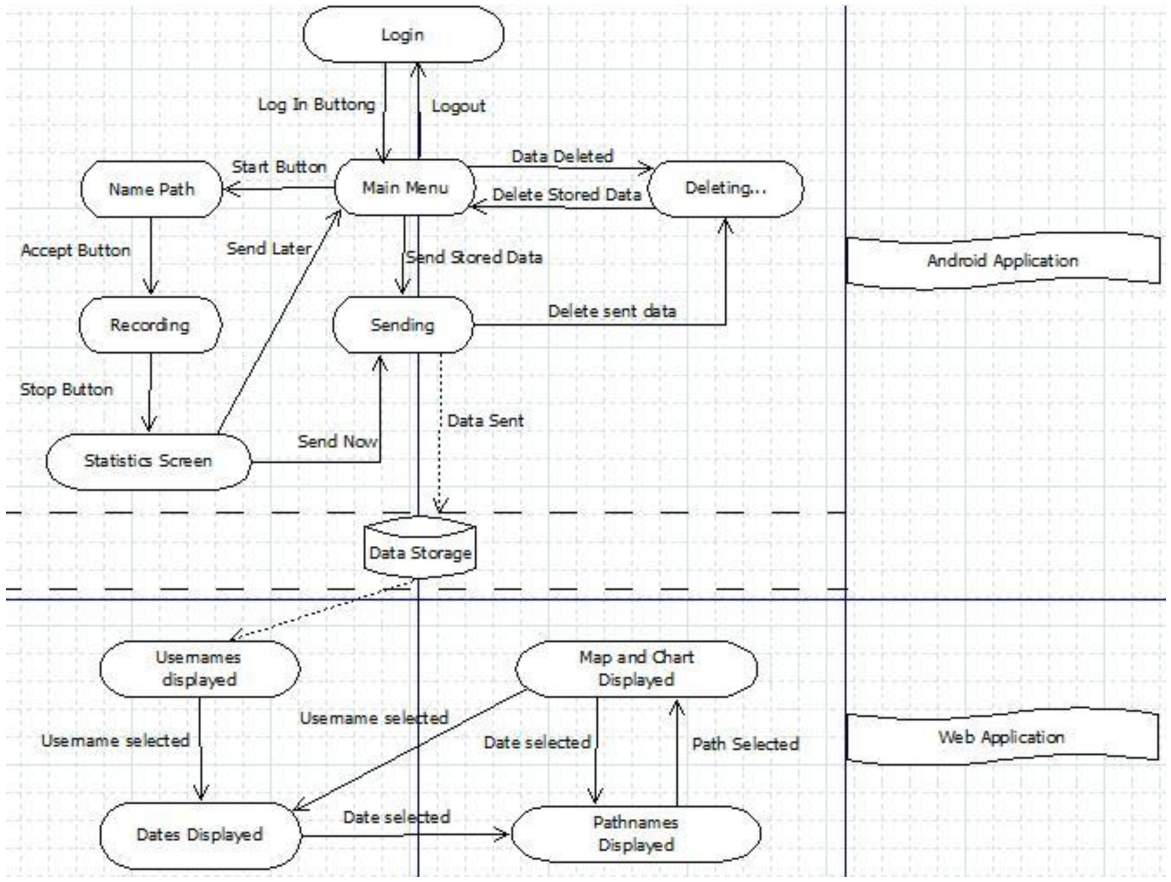
Newmont Mining Corporation
Colorado School of Mines

Figure 5.2: Bump-O-Meter Website (after Username is selected)



Newmont Mining Corporation
Colorado School of Mines

Figure 6: Bump-O-Meter Flowchart


Login
Log In Buttong
Logout
Name Path
Start Button
Main Menu
Data Deleted
Delete Stored Data
Deleting...
Accept Button
Send Later
Send Stored Data
Android Application
Recording
Sending
Delete sent data
Stop Button
Data Sent
Statistics Screen
Send Now
Data Storage
Usernames displayed
Map and Chart Displayed
Username selected
Username selected
Date selected
Path Selected
Web Application
Dates Displayed
Date selected
Pathnames Displayed

Design and Implementation Decisions

Decisions

- The database schema on the phone is different than that of the schema on the server as a result of several considerations. Originally we considered having the same schemas for both databases, but this resulted in complicated data transfer. We then considered having only 1 table in the phone's database. This resulted in a table that was difficult to query. We settled on a 2 table schema on the phone because it minimized the amount of data being stored while allowing for simple queries and easy data transfer.
- The website objective was to have the simplest user-friendly interface. The website uses AJAX because the web page only has to manipulate a small set of interfaces. If AJAX was not used, the web page would frequently have to be reloaded, and user data would have to be stored in an inefficient manner. The goal was to have the user select options from a set of drop-down menus without having to reload the page. Using AJAX prevents the user from requesting data that does not exist, and makes the GUI elegantly simple.
    - In order to use AJAX, each change of a drop-down menu makes a database request to the server in PHP, which is then returned to the client in JSON. The client-side Javascript then parses the JSON, and appends the retrieved data onto the drop-down menus.
- Google Chart Tools was chosen due to ease of use, clear documentation, and reputation in the web development community. Using it was straight forward and easy to use with AJAX.
- A few Javascript mapping libraries were recommended by the client such as Openlayers, but after further research, Google Maps API was chosen for the same reasons as Google Chart Tools. Also, Google Maps is a well known map interface and is familiar to many web users.
- The PostgreSQL JDBC driver was chosen due to ease of use and reputation in the Android development community. Other methods were complicated and unnecessary.
- The slowest sensor setting was chosen for the accelerometer because the other two settings provided too much data which could potentially create problems for data storage on the phone. Also, such an accuracy was not needed due to the speed of bicycles.

Issues

- GPS accuracy was a major issue that could not be resolved. This led to an inaccurate representation of the data collected. Upon doing further research, we learned that the GPS installed in the smart phones is a low grade chip. This was not an issue that we could have easily resolved. The most simple way of solving this issue would be to use a GPS specific device.
- We encountered another minor issue related to transferring data. When trying to send the data from the phone to the server, the phone would freeze until the data was sent. This was fixed by stalling the data transfer until after changing screens so that the user would know what is happening.

Results

Before the completion of this application, there was nothing on the market that does quite what Bump-O-Meter does. No app existed that tracked paths and their bumpy spots and displayed this data to a web page. Now a community of riders can track where they have ridden and identify potential trouble spots.

Despite some minor drawbacks technical issues we met all functional and non-functional requirements for this project. Given more time however, these are a few goals that could be implemented to improve the application:

Areas for expansion:

- Speed up transfer of data
- Improvement of GPS accuracy through path prediction
- Implement a sending screen that does not freeze and displays until completion of data transfer
- Add a "Settings" screen
- Expand on the already existing statistics screen at the end of rides
- Add an option that displays a map to the user during their ride
- Add an alert system that would notify the rider a bump was coming up
- Statistical analysis of frequently ridden paths to better identify trouble spots
- Secure Log-in system

Resources

Google Charts Library Documentation

- https://developers.google.com/chart/

Google Maps Library Documentation

- https://developers.google.com/maps/

Android Development Website

- http://developer.android.com/

PostgreSQL Library Documentation

- http://www.slideshare.net/markwkm/andoird-postgresql

Android Tutorials for Java

- http://www.vogella.com/