

Software Design Specification

Berry Bus Application



April 30, 2013

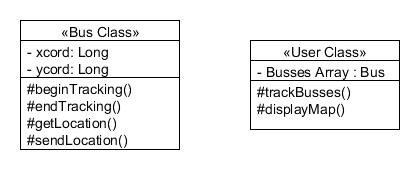
Berry College

Dr. Nadeem Hamid

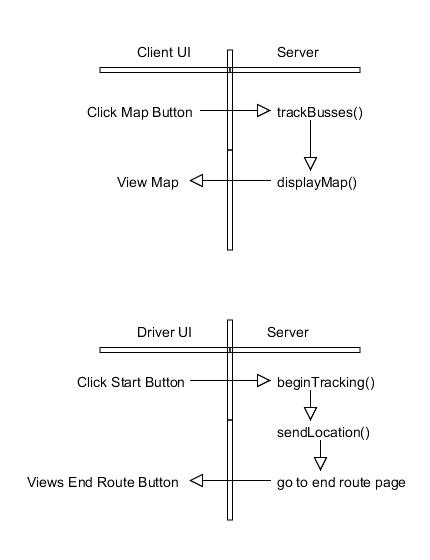
Adam Gillfillan, Cal Supik

## System Architecture

Our system will consist of a few modules that will work in unison to allow input from a ‘driver’ and provide output to a ‘client’. Some of the major components that we have to take into consideration is tracking a bus, and displaying the bus on a map in real time. We also have to identify what data we will store and how we will be storing it. For this project, the data that we will be most interested in is the GPS coordinates of the buses. As a group, we have decided the best way to store and access this data is through a PHP webpage. This PHP webpage will be receiving coordinates of the buses while they are being tracked and will also be sending/providing these coordinates to other components of our system. The other side of our project is the client side. On this side we will be running a program that will be continuously retrieving these coordinates and displaying them on a map for the user to see.



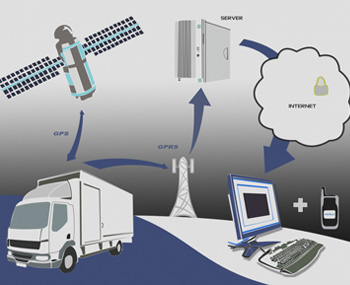
The image above describes the classes that we will be using in this project. As you can see we listed possible fields in each class, and possible functions. These class diagrams are not definite or completed to the fullest but more or less an outline or guide.



The image above shows some possible sequences of events that will occur on the 'driver side', 'client side' and server side of this project. These sequence diagrams are not totally specific but merely pseudo code of what we plan to implement.

## Tools

To complete our project, a few tools will be needed. These tools include HTML5, JavaScript, PHP, and iWebKit. HTML5 is the latest HTML standard, following its predecessor HTML4. The reason we will be using HTML5 is the fact that it will let us access the GPS coordinates of the cell phone that will be placed on the buses. It will also make our web application both look and feel like a native application. JavaScript is needed to allow the maps to be displayed on the application and will also show the bus moving along it's route. As mentioned above, we will be using PHP to access the coordinates to a web page. This page will be updated as frequently as desired, while trying to keep the runtime as low as possible. Though there might be a slight delay, we intend on making the app as accurate as possible. iWebKit is a program that is intended for people to write web applications for iPhones. It lets the user edit the framework to their desired specifications from start to finish. It lets the user customize different elements of the application and has the ability of integrating Google maps into the app. These features describe why we chose to use this program.



The image above shows the "behind the scenes" process of how our project is actually going to work. The buses GPS coordinates are tracked via satellite. These coordinates are then sent to a server and saved on a web page. The application will then fetch these coordinates and display a map of the moving bus.

## Process

#### Release Goals:

Release 1.0: Getting and sending GPS coordinates of the bus work to and from the server. This includes getting/posting to a PHP page, in which the GPS coordinates from the phone will be sent to a PHP webpage. These coordinates will then be stored on a database. Tracking of the buses in a real-time display work. UI is basic and only a simple link to “map” to track the buses. When this link is clicked, a map will then be displayed of the current GPS coordinates being pulled from the database. First release is able to display just one bus. Server is set up and communication between bus, server, and user is fluid and functional.

Testing: The user with the “bus” phone will sit in a car while someone else drives him around. One of us will monitor if the sending of the GPS coordinates from the “bus” phone to the server works correctly. This sequence will be repeated an estimated 10 times.

Milestones:

1) Send GPS coordinates from a phone.

2) Create PHP page to store GPS coordinates sent from phone.

3) Create basic and minimalist UI for web application.

4) Get a bus to display on the map of the web application.

5) Get the bus to display its real-time position on the map of the web application.

Release 2.0: UI design is improved and in its final state. Map now displays any number of buses that are sending their GPS coordinates to the server.

Testing: The users with the “bus” phones will sit in two separate cars while someone else drives each person around. One of us will monitor if the sending of the GPS coordinates from each “bus” phone to the server works correctly. This step is crucial in ensuring the data is readable on the PHP page storing this data.

Milestones:

1) “Tag” the data sent from each “bus” phone to distinguish it from a different bus.

2) Test each phone’s ability to send coordinates.

3) Create a visually improved UI for the web application.

Release 3.0: “Pick Me Up” feature is released.

Testing: Much time will be spent with the security of this feature—ensuring a user who has requested a pick up and has been picked up no longer displays to the driver as requesting a pick up.

Milestones:

1) The user’s phone correctly sends GPS coordinates to a PHP page on our server.

2) Each bus phone correctly displays on map the user requesting a pick up.

3) Conquer security issue #1: a user who has requested a pick up and has been picked up no longer displays to the driver as “requesting a pick up”.

4) Conquer security issue #2: a user who has requested a pick up and later decides to not be picked up no longer displays to the driver as “requesting a pick up”.

5) Prompts the user to proceed to a “waypoint/bus stop” before their GPS coordinate can be sent to a bus driver, to ensure that no one can request a pick up while in unreachable or remote locations.