Live Map of the DC Bus System: Design

Ian Will, Jason Cluck

Abstract

The Washington Metropolitan Area Transit Authority (WMATA) publishes an API providing access to live bus positions and route details. There are a few iOS, Android, and Web applications that use this data to help make transit riding more pleasant. For example, the NextBus application shows the expected arrival time of the next bus for a given route and stop.

However, all current applications are missing some functionality that limit their usefullness. None of these applications show all bus routes simultaneously on a map. This limits the transit rider's ability to discover alternate routes. These current applications also do not take advantage of showing the live bus positions on the map. A combination of these two points of information would be very helpful for a bus rider. If the desired bus requires a long wait, a short walk to a nearby station on another route with an approaching bus may provide a better alternative. This application aims to let the individual riders make informed decisions using all of the available information.

We will use the WMATA developer API to retrieve route details and bus positions from WMATA's servers at regular intervals. We will then use the Google Maps API to display the route data, updated in real time. The WMATA developer API places restrictions on frequency of data access–five times per second or 10,000 times per day maximum. Those limits prohibit accessing the data every time our web page is loaded (were it to be widely used); therefore our application will poll the WMATA servers roughly once every 5 seconds and serve the latest snapshot from our separate web server. We will use the Ruby on Rails web framework to build a web server that maintains a database of route and position data, updates from WMATA's servers periodically, and displays the data on a map efficiently. The application will be deployed using the Heroku cloud application platform.

Task Breakdown

1. Initial rails setup [Cluck]
2. CSS styling using twitter bootstrap [Cluck]
3. Display map on index page [Cluck]
4. Fetch bus positions from WMATA servers [Will]
5. Store WMATA response in database [Will]
6. Draw markers on map for bus locations [Will]
7. Draw lines on map for bus routes [Will]
8. Show info window when bus marker is clicked, including the following
    (a) Route name
    (b) Schedule deviation (lateness)
    (c) Time of last position update (data staleness)
    (d) Headsign
    (e) Direction
9. Display stops on map
10. Control WMATA polling to avoid exceeding usage limits
11. Display bus stops with markers on map
12. Show info window when stop marker is clicked
    (a) Show which routes and directions the use the stop
    (b) Show next bus wait time projections for each route that uses the stop
13. Fetch routes from WMATA servers
14. Store routes in database
15. Fetch stops from WMATA servers

16. Store stops in database
17. Layer toggle widget that shows Google traffic overlay, bus markers, and stop markers

Diagrams

Figure 1: System Overview

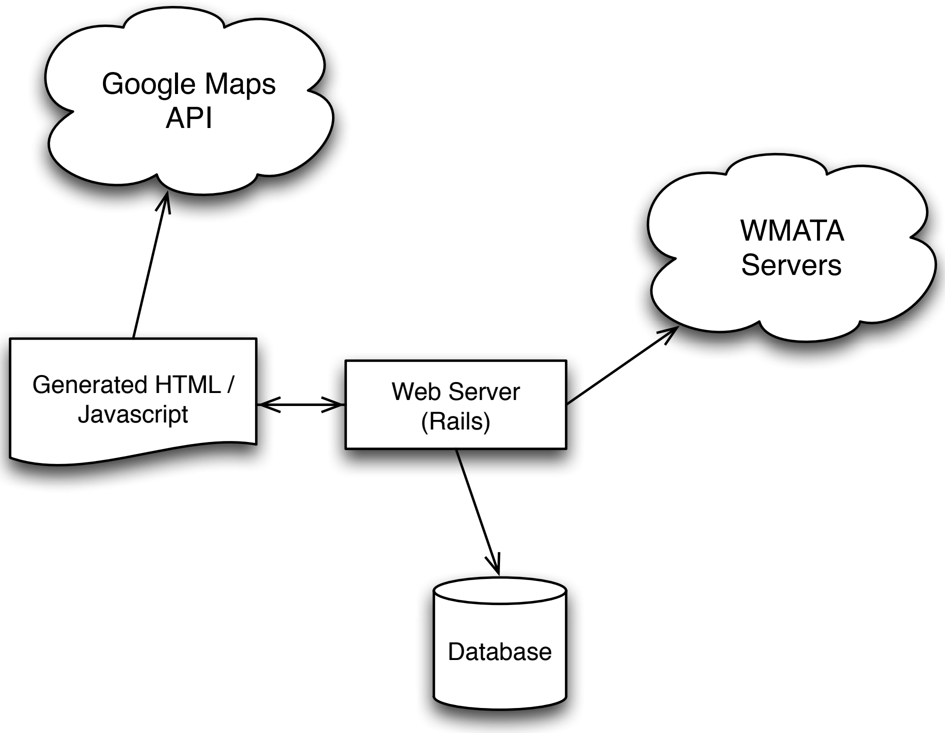


Figure 2: Class Diagram

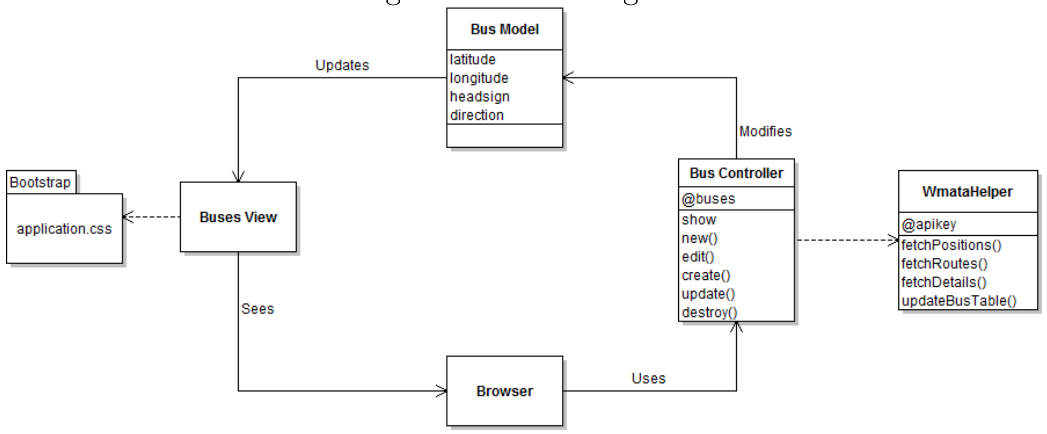


Figure 3: Entity Relationship Diagram

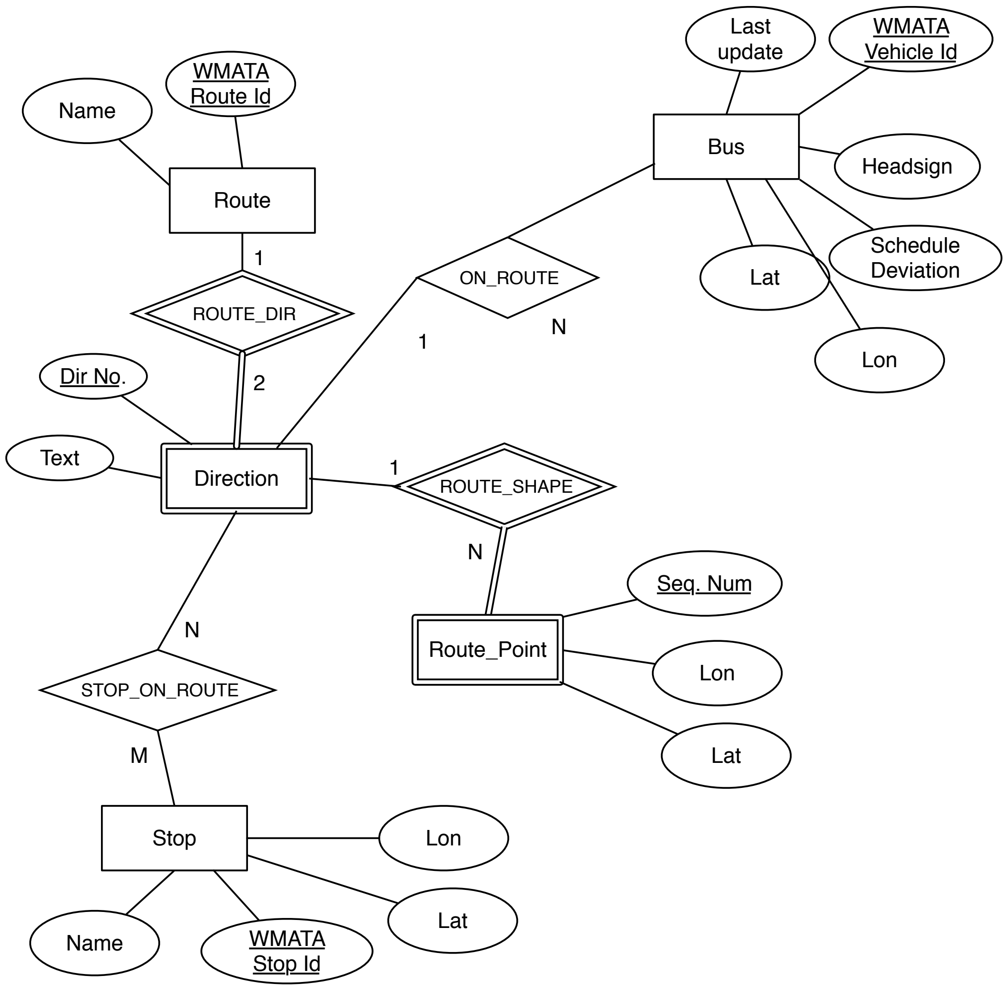


Figure 4: Database Schema

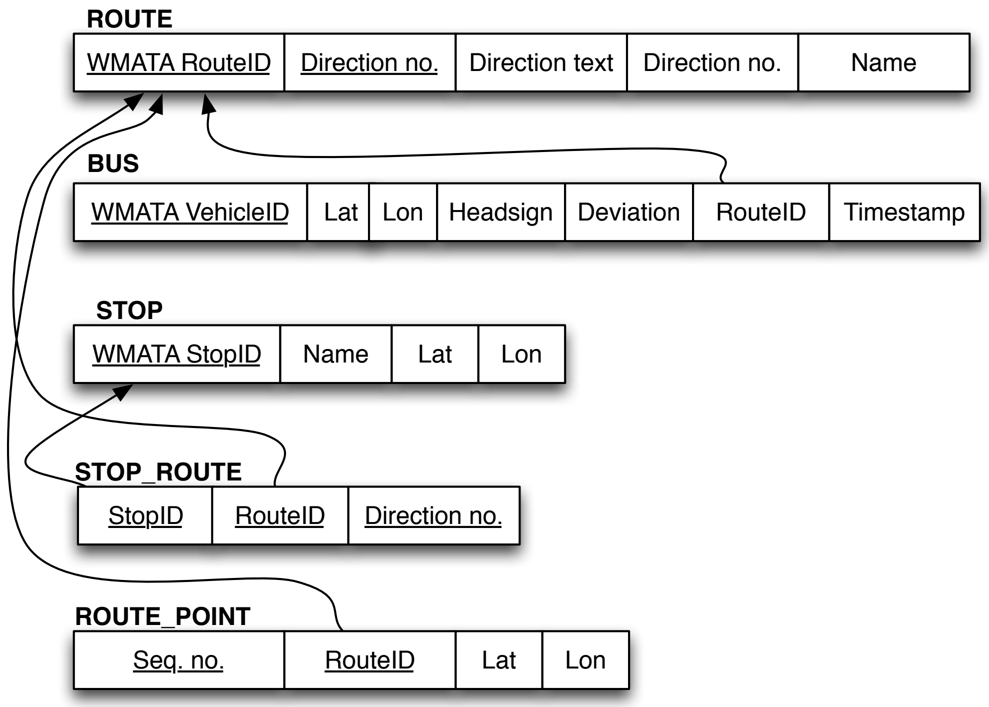


Figure 5: Sequence Overview

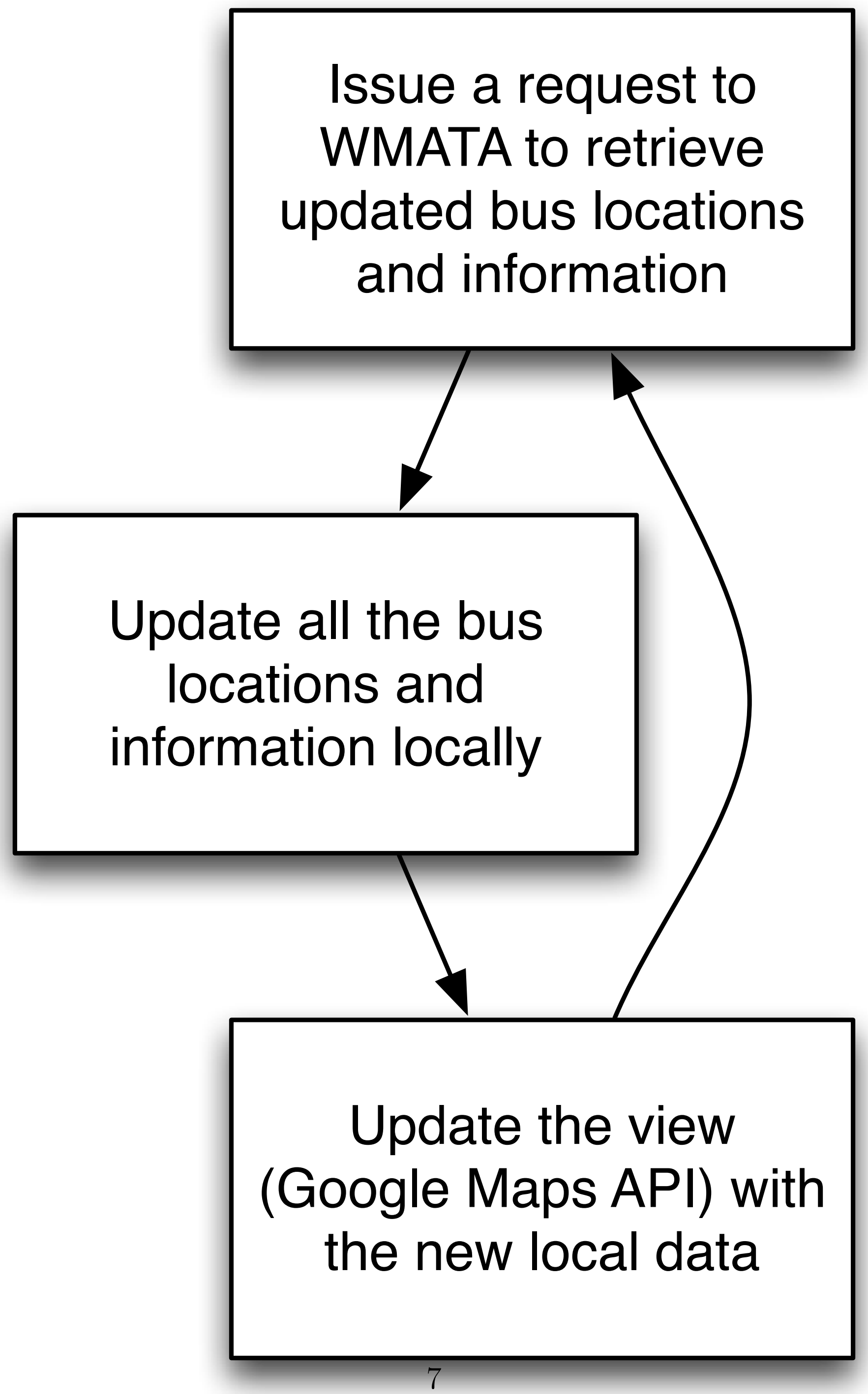
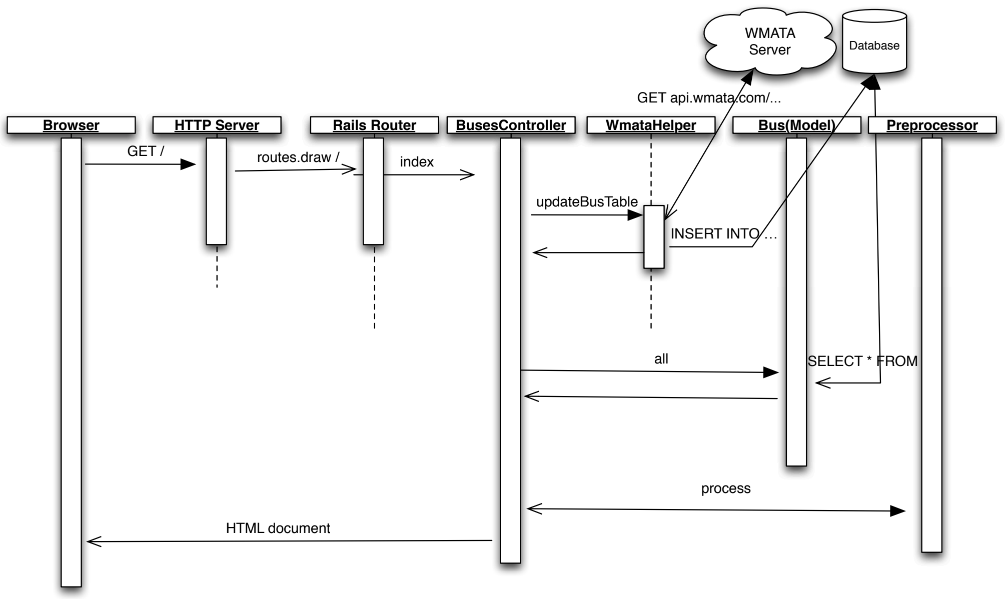

Issue a request to WMATA to retrieve updated bus locations and information
Update all the bus locations and information locally
Update the view (Google Maps API) with the new local data

Figure 6: Rails Request Sequence Diagram


WMATA Server
Database
GET api.wmata.com/...
Browser
HTTP Server
Rails Router
BusesController
WmataHelper
Bus(Model)
Preprocessor
GET /
routes.draw /
index
updateBusTable
INSERT INTO ...
all
SELECT * FROM
process
HTML document