Course Title: 14:332:452 Software Engineering

Group 4

Onward (Traffic Monitoring)

Report 2 Part 1

February 26, 2017

GitHub (e-archive): https://github.com/solejar/traffic_ru_ece
Website: http://www.onwardtraffic.com/

Team Members

Name	Email
Ridwan Khan	ridwankhan101@gmail.com
Brian Monticello	b.monticello23@gmail.com
Mhammed Alhayek	almoalhayek@gmail.com
Sean Olejar	solejar236@gmail.com
Lauren Williams	laurenwilliams517@gmail.com
Shubhra Paradkar	shubhra.paradkar@gmail.com

Individual Contributions Breakdown

All team members contributed equally.

Table of Contents

Interaction Diagrams

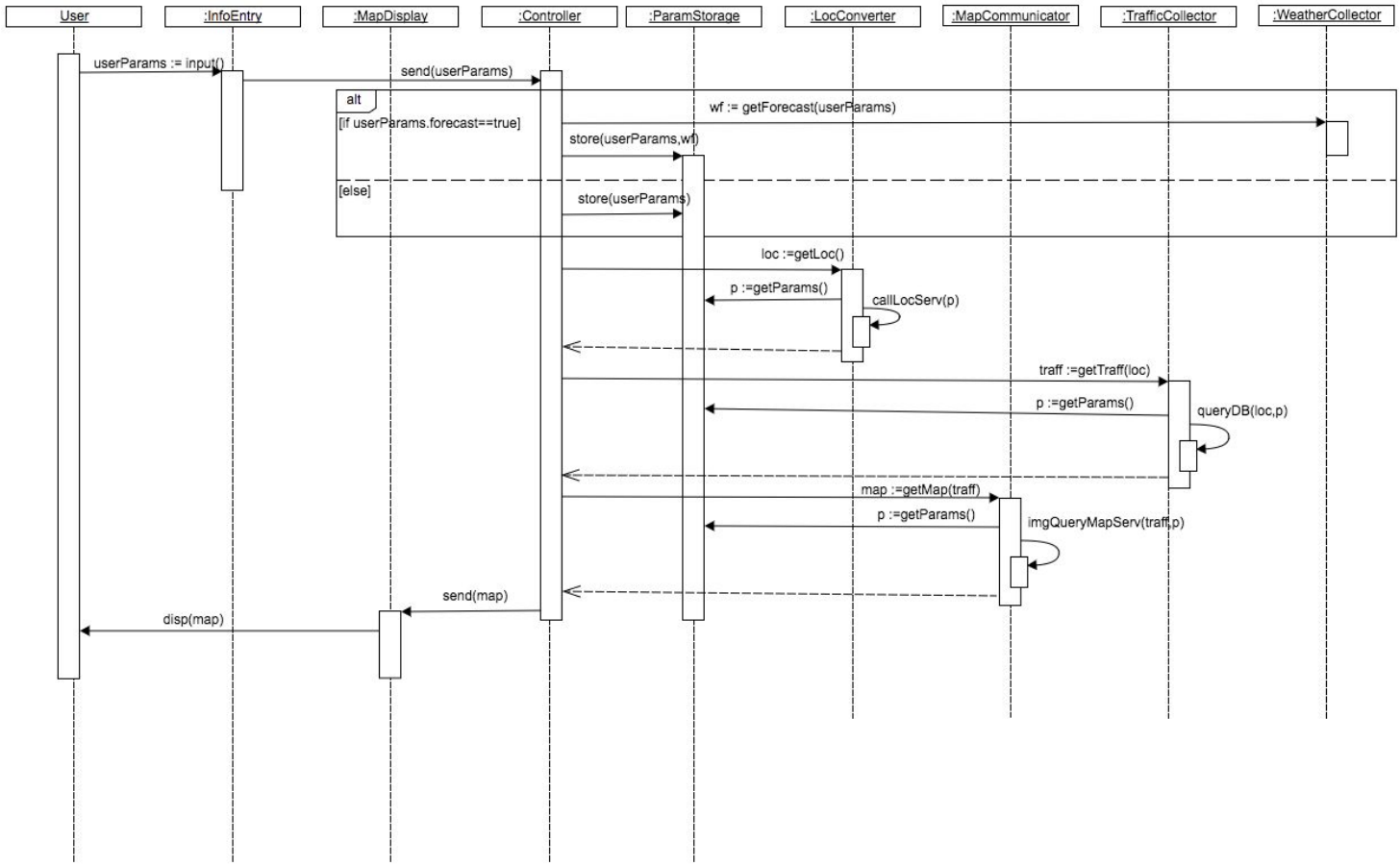

Figure 1: Sequence Diagram of UC-1 with extension UC-6

This diagram incorporates both UC-1, the “Heat Map” feature, and its extension UC-6. UC-6 is simply the situation in which the user decides to use forecasted weather as opposed to entering it in manually, therefore UC-6 does not have its own sequence diagram. This diagram is initiated from after the point where the user chooses the “Heat Map” feature on the landing page and begins to input parameters. We designed all of our concepts with cohesion and modularity in mind. Having all communication mediated by the Controller entity allows us to mostly decouple the individual components of our system. This helps keep our system modular, since we can add and remove components at will without risk of disturbing the functioning of other components. Additionally, each component has a focused, cohesive purpose. This sequence diagram is directly derived from our domain model.

As described in our alternate design decisions, we originally had an issue distinguishing in our system between the different parameters of UC-1 and UC-6. In order to clarify the origin of the parameters, we added the entity ParamStorage, which handles the job of distinguishing and storing the different parameters for the different use cases. Also, we decided against explicitly including the participating actors in the diagram to improve the simplicity and readability of the sequence diagram. Our participating actors are the Database, Location Service, Mapping Service and Weather Service. The system is client-side only when the user is entering information and viewing the map. Everything else happens on the server-side.

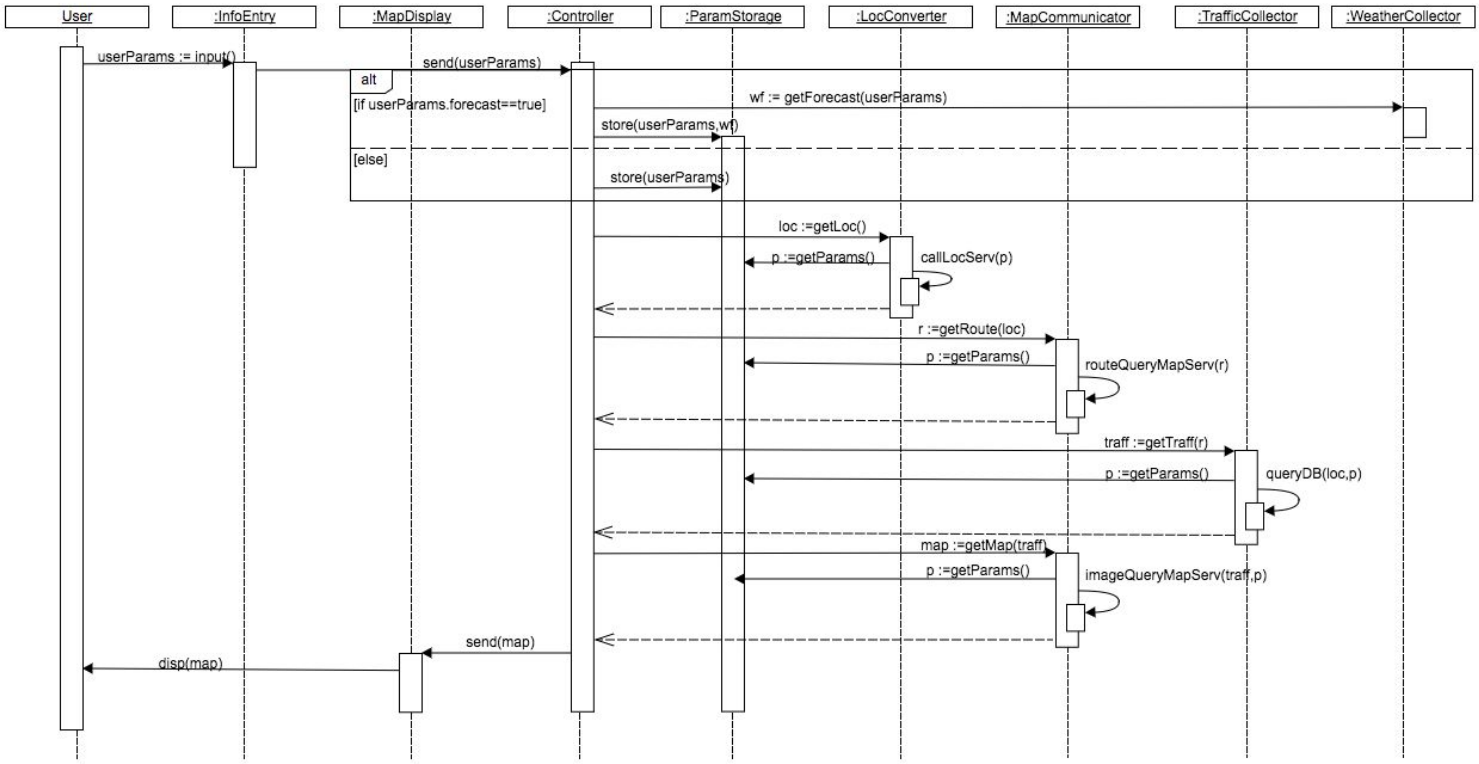

Figure 2: Sequence Diagram of UC-2 with extension UC-6

This diagram incorporates both UC-2, the “Route” feature, and its extension UC-6. This diagram is similar to Figure 1 above. The components used are all the same, though the order in which they are accessed is different as well as the function calls that are required to implement the route feature. As a result, the design principles mentioned in the description for Figure 1 still apply here. This diagram is initiated from after the point the user chooses the “Route” feature on the landing page and begins to input parameters.Both use cases use the same conceptual objects, which is by design to make our concepts reusable since both features have similar functions.

Alternative Design

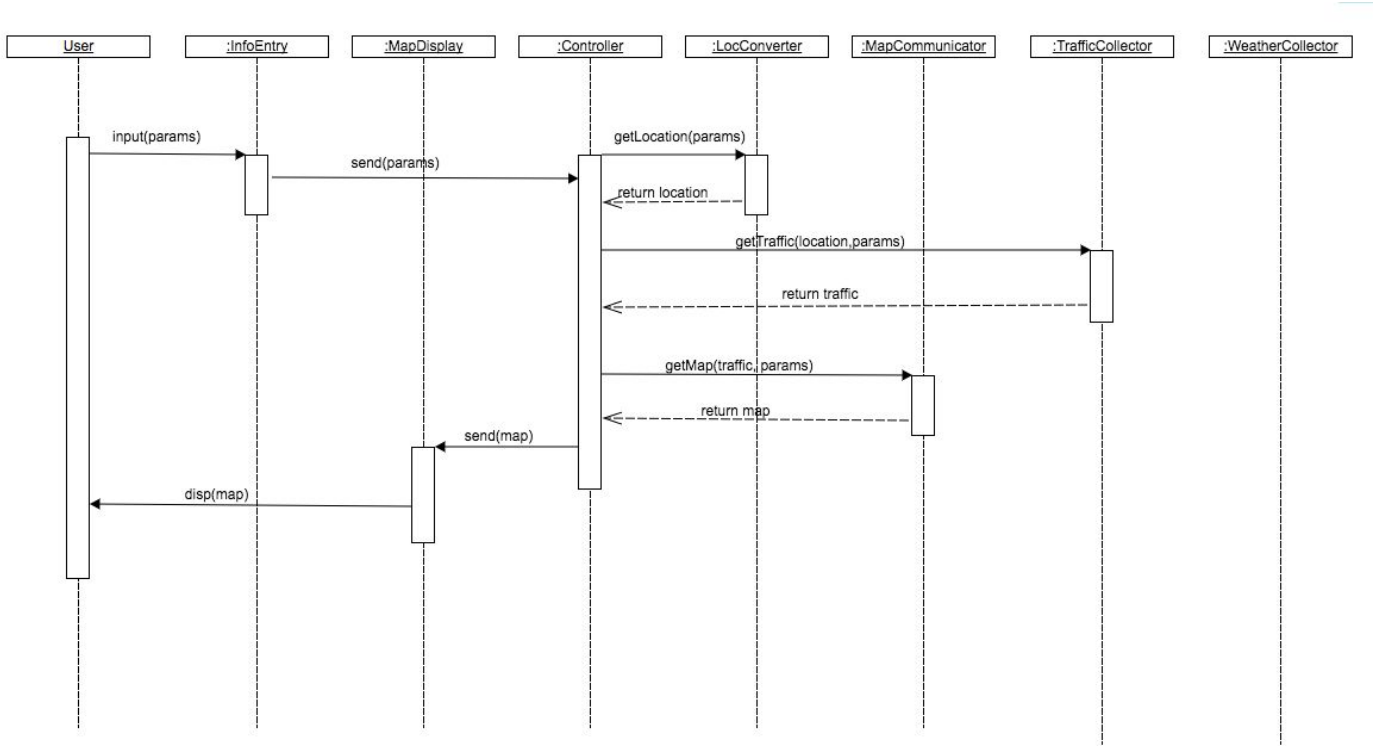

Figure 3: Alternate Design Diagram

This diagram was the first design we came up with for the UC-1 interaction diagram. In this diagram, InfoEntry sends the user inputted parameters to the controller, which stores these parameters for use by the other components. We decided against this because we felt that it was non-cohesive to have the Controller concept also in charge of info storage. This is shown in our final design for the interaction diagram (Figure 1), where the concept ParamStorage receives the user inputs and stores them. We also found this necessary because not all of the parameters that are used come from the user, such as in UC-6. The addition of the ParamStorage concept allows us to resolve parameters of different origin into one easy-to-access location. The final design is, as a result, more cohesive because the Controller is now no longer responsible for keeping track of parameter info.

Project Management

Merging the Contributions from Individual Team Members

While collaborating on the report it sometimes was difficult to blend writing styles of all six group members. Often times several different ways of writing would conflict and interrupt the cohesiveness and coherence of the section or sections of the report. In order to cooperate all the ideas of the group, it was often more beneficial to let all members write their ideas and then later work to edit the document so that it would smoothly transition from one response to the next. The group took turns editing the document at different moments so that each member could check the coherency of the document as it came together. Another problem that we encountered was the different conventions for labeling figures and diagrams and its respective elements. We dealt with this issue by coming to common conventions of detailing figures and diagrams that would most effectively work for the group in both the reports and in the implementation phase. By doing this we were able to ensure consistency with how we communicated our ideas to those reading our report.

Project Coordination and Progress Report

Currently the group is implementing UC-1 and UC-2. The group is working on tackling these two use cases are currently being worked on by individual subteams 1 and 2. Each subteam has split the priorities and responsibilities for these two use cases. The database used for data collection has already been created and is functional in terms of properly collecting data for weather and traffic incidents. This database was created by all members of the group and will be later used properly in the use cases. Some relevant project management activities include keeping a schedule that accounts for the report submission dates, the demo dates, and necessary implementation due dates. This schedule allows for every group member to aware of the specific work that needs to be completed by a specific date so that each group member can plan his/her schedule and complete the work by the designated time.

Plan of Work

Our team is working according to the Gantt Chart below.

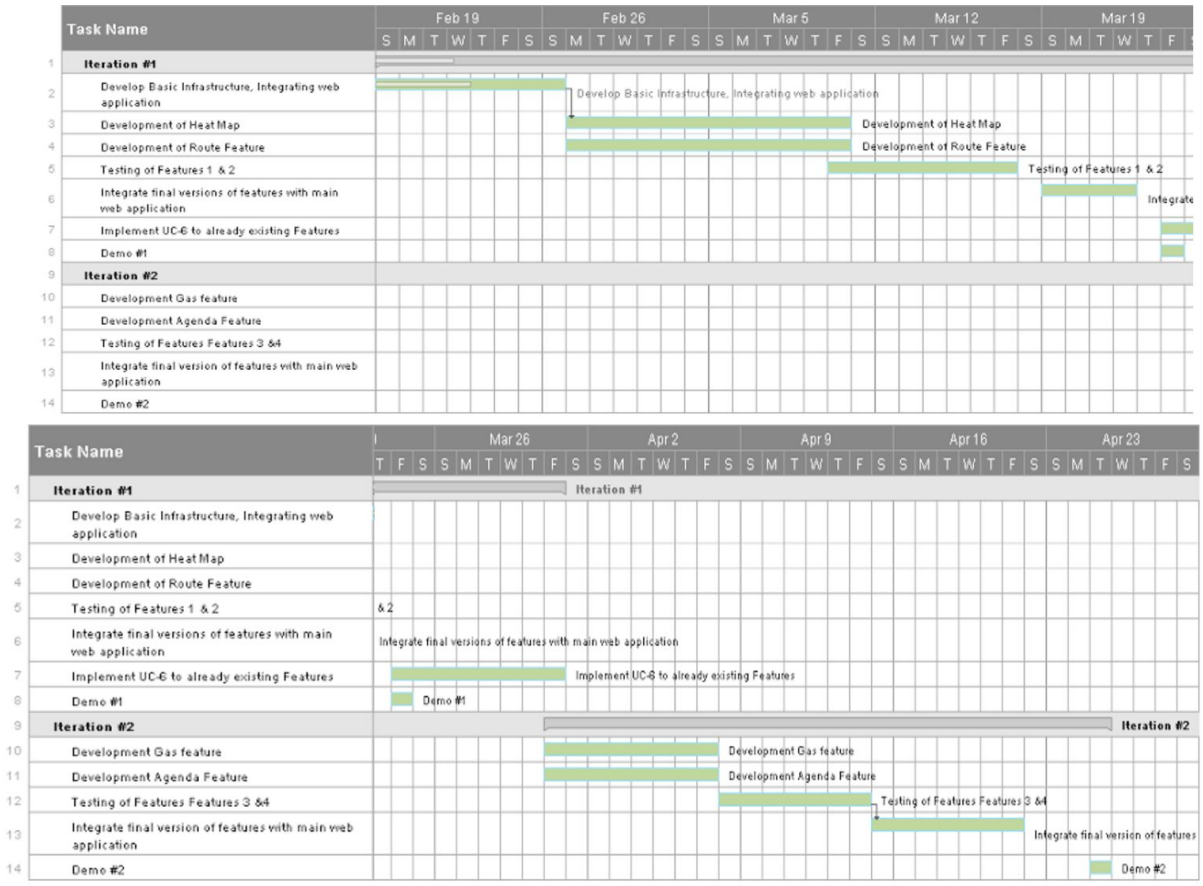

Figure 4: Gantt Chart

Breakdown of Responsibilities

Heat Map Feature:
The Heat map was discussed and the features within it were formulated and thoroughly worked on by all members of the group. The development of this feature will be split up evenly between Brian, Ridwan and Mhammed. They will be equally responsible for coding and testing this feature. It is currently premature to say which specific modules and classes will be worked on by each member.

Route Feature:
The Route Feature was discussed and the features within it were formulated and thoroughly worked on by all members of the group. The development of this feature will be split up evenly

between Lauren, Shubhra, and Sean. They will be equally responsible for coding and testing this feature. It is currently premature to say which specific modules and classes will be worked on by each member.

Sean will be coordinating the integration of the two features. He is the one who approves pull and merge requests from the branches. He will be the one with the ownership of the master branch and make sure that both the Heat Map and Route branches are merged to it.

Each individual team will test to make sure that their feature works in the program as a whole. Once the Heat Map Feature and the Route Feature are fully implemented, it will then be Ridwan's task to test whether the two parts work together as a cohesive unit. Each team will perform the integration tests for the first demonstration, and Ridwan will complete integration tests on the web platform as a whole. These integration tests will also be completed as each team completes the various aspects of their implementation. Since both of these features have similar methods of implementations, and as a result have overlapping concepts, both sub teams will also work to make sure that their contributions to the concepts/classes do not interfere and cause merge conflicts. The sub teams will test how each aspect of the project integrates with the project as a whole ie. with the web platform. However, we would also like to run integration tests on both features once completed and how these features act under user interaction with the web platform.

References

- Bing Traffic API - *Bing Developer Network.* 2017. <https://msdn.microsoft.com/en-us/library/hh441725.aspx>
- Current Weather Data - *Weather Underground API*. 2017. <https://www.wunderground.com/weather/api/>
- myGasFeed API - *JGSolutions.* 2010. <http://www.mygasfeed.com/keys/api>

The above references are the current API's being used to collect data and that will be used for future implementation.