Design Document

Kevin Niemiller

Use Case 2: Update Data by System

This diagram was created to show how the system updates Road Activity from the ODOT website. The system runs a loop every 5 minutes, and runs this loop forever. Everytime the loop executes, it executes an ImportData() routine which does a request to the ODOT website to pull the XML file in. This XML file includes all Road Activity. This design uses the Low Coupling GRASP pattern. It uses low coupling because the system does not care what information or the amount of information is being set from ODOT; it only cares that it is an XML so that the system can parse through this information.

Use Case 4: Analyze Road Data

This diagram was created to show how the system allows the user to analyze road activity. The user navigates to the Road Activity page which shows all road activity. The user can then set filters for the category type, start date, and end date. Once the user selects these, the Analyze Road Activity sends this request to Road Activity who then sends back the roads (and road information) of what was selected. This diagram is based off of the Information Expert. It uses information expert because the Analyze Road Activity is getting\relying on the information from Road Activity who has all the information.

Use Case 5: Enter Roads Travelled

This diagram was created to show the relationship between TravelPaths, AddTravelPath, RoadName, and TravelPath. The TravelPaths shows all roads that have been added by the current user. The user makes an AddTravelPath request which looks up all possible roads to display to the user. Once the AddTravelPath information is inputted, then this information is submitted by the user which makes AddTravelPath create a TravelPath (with the road information). From here, this TravelPath (and it's information) can be displayed on the TravelPaths for that user. This diagram uses multiple GRASP patterns. This design uses High Cohesion because multiple steps through this Use Case rely on other classes and information within those classes. This design also uses the Creator pattern, because AddTravelPath creates a travel path that houses the new path and all the paths information (such as start mile, end mile, start time, end time, days of the week, etc.). This design also uses Information Expert since it relies on getting the Road Names so the user can pick one of these roads to enter as a road/path.

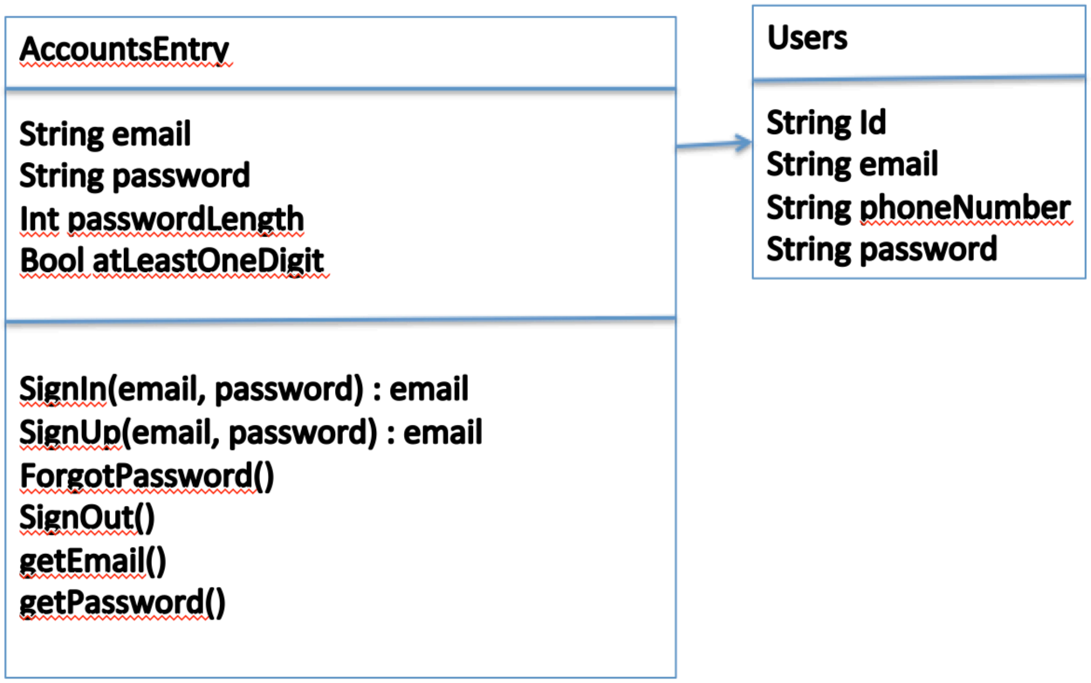
Use Case 14: View all Travel Paths

This diagram was created to show how the user interacts with the TravelPaths. The user navigates to the TravelPaths page and is able to view all travel paths that the user has entered. This design uses the Controller GRASP pattern. It uses this pattern because the user will select a page from the view and then the controller grants that request by loading the TravelPaths page along with all the travel paths for that user.

Use Case 15: Edit Travel Paths

This diagram was created to show the relationship between TravelPaths, EditTravelPath, RoadName, and TravelPath. The TravelPaths shows all paths that have been added by the current user. The user makes an EditTravelPath request which looks up all possible paths to display to the user. Once the EditTravelPath information is edited, then this information is submitted by the user which makes EditRoad update the TravelPath (with the road information). From here, this TravelPath (with it's information) can be displayed on the TravelPaths for that user. This diagram uses multiple GRASP patterns. This design uses High Cohesion because multiple steps through this Use Case rely on other classes and information within those classes. This design also uses Information Expert since it relies on getting the Road Names so the user can pick one of these roads to enter as a road/path if they are editing this information and also to call up the travel path the user is wishing to edit so that this information can be pre-populated to make it easier for the user to modify.

CLASS DIAGRAM Accounts-Entry and Users



CLASS DIAGRAM Update Data By User, Update Data by System, and Current Road Activities

Session

String key
String value

get(key): value
set(key, value)

ImportData

File xml
String url

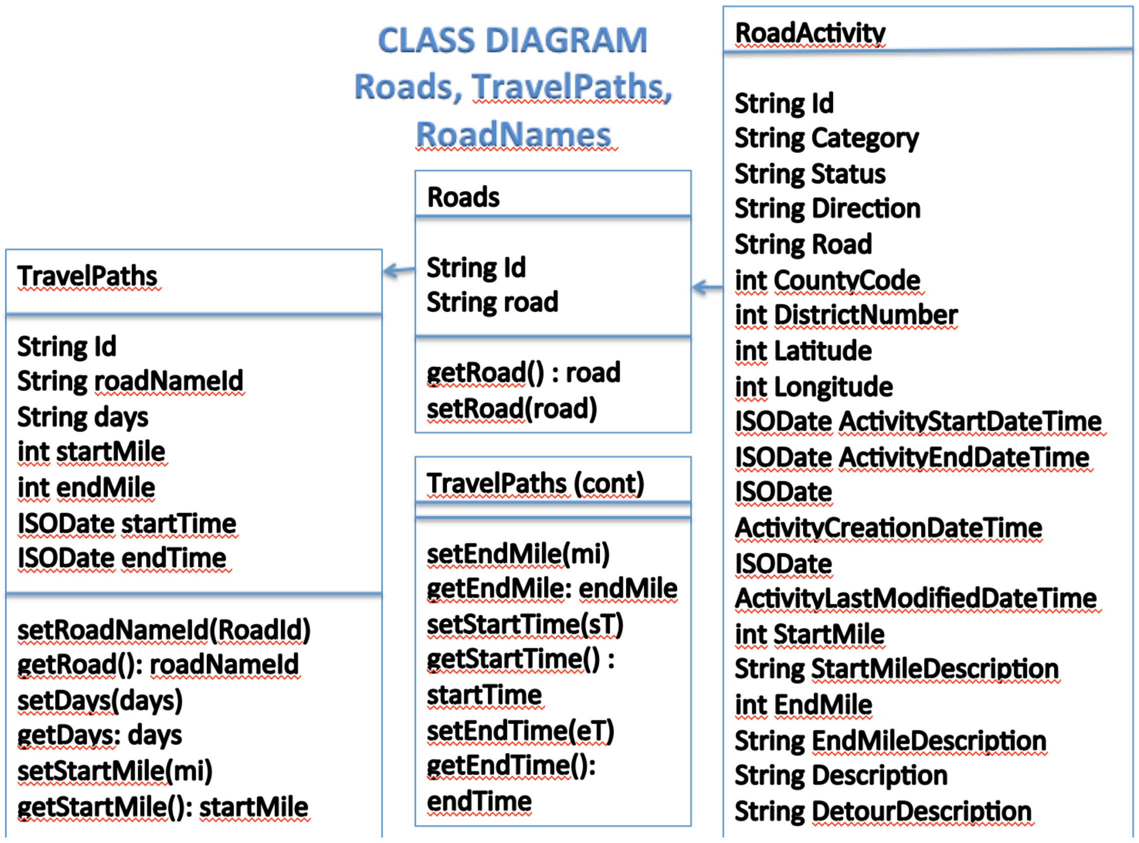
makeHttpRequest()
getXML(): xml

Data

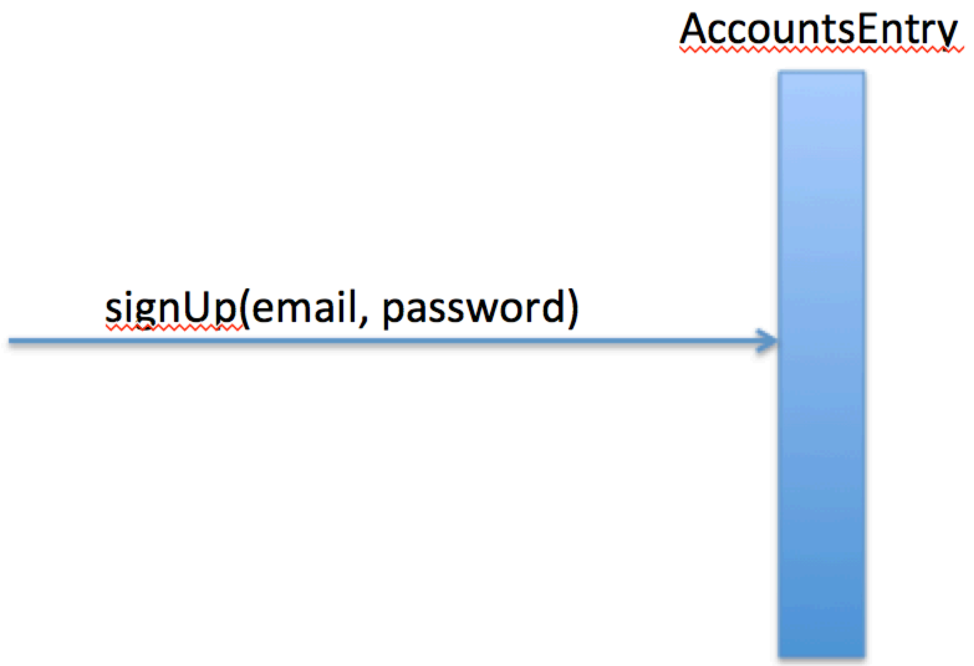
File xml

importData(xml)
filter(filter)

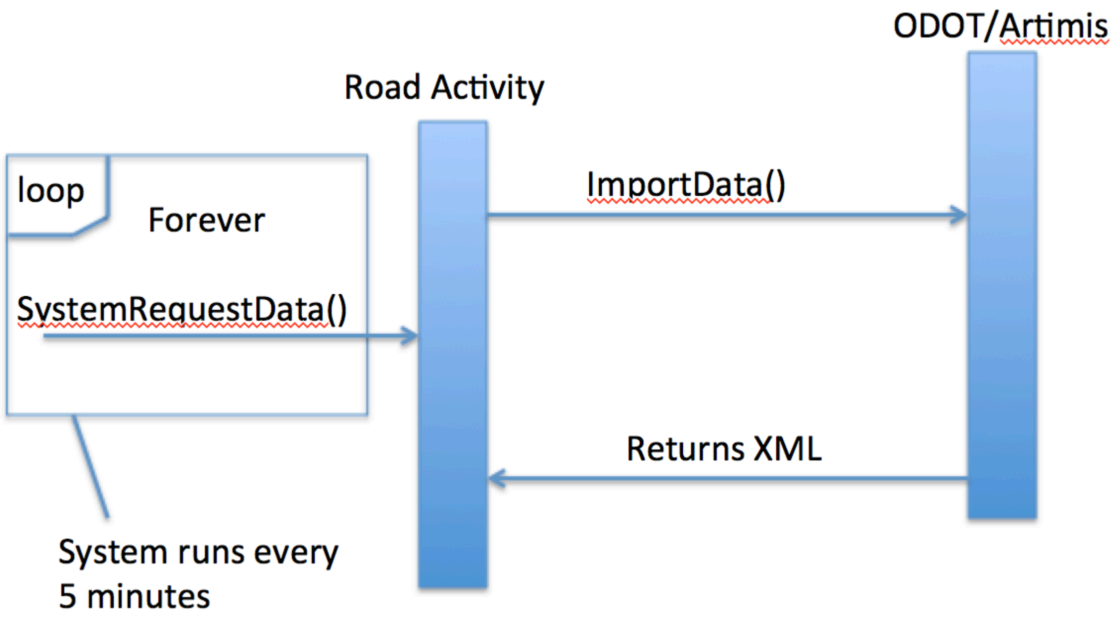
CLASS DIAGRAM Roads, TravelPaths, RoadNames



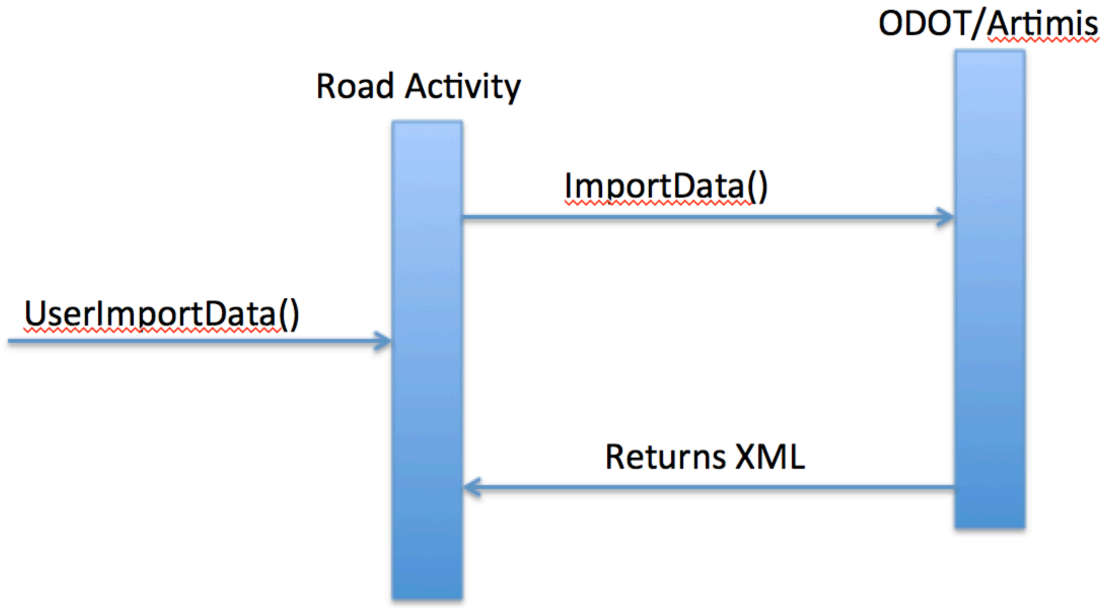
Interaction (Sequence) Diagram – Use Case 1: Sign Up



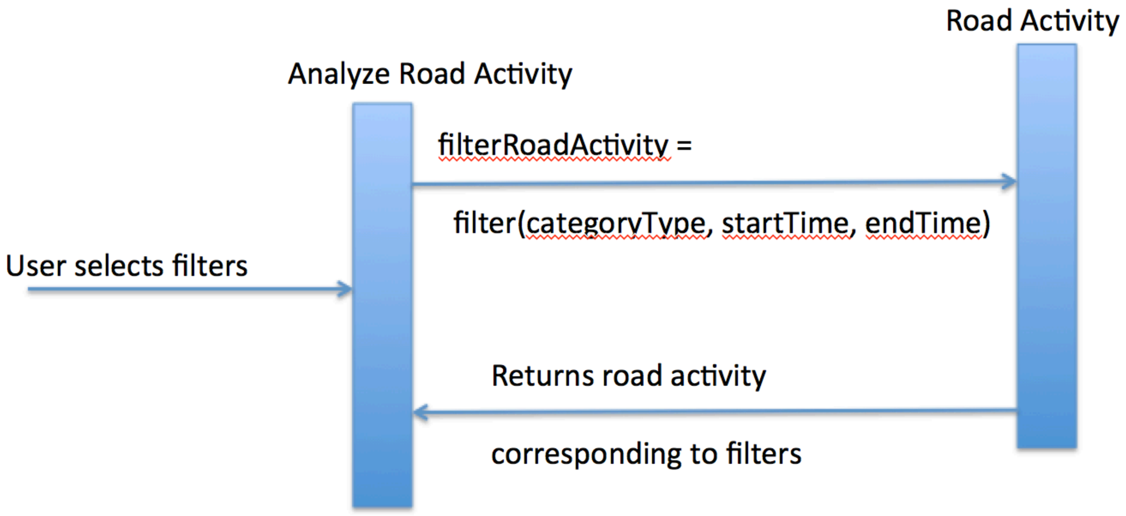
Interaction (Sequence) Diagram – Use Case 2: Update Data by System



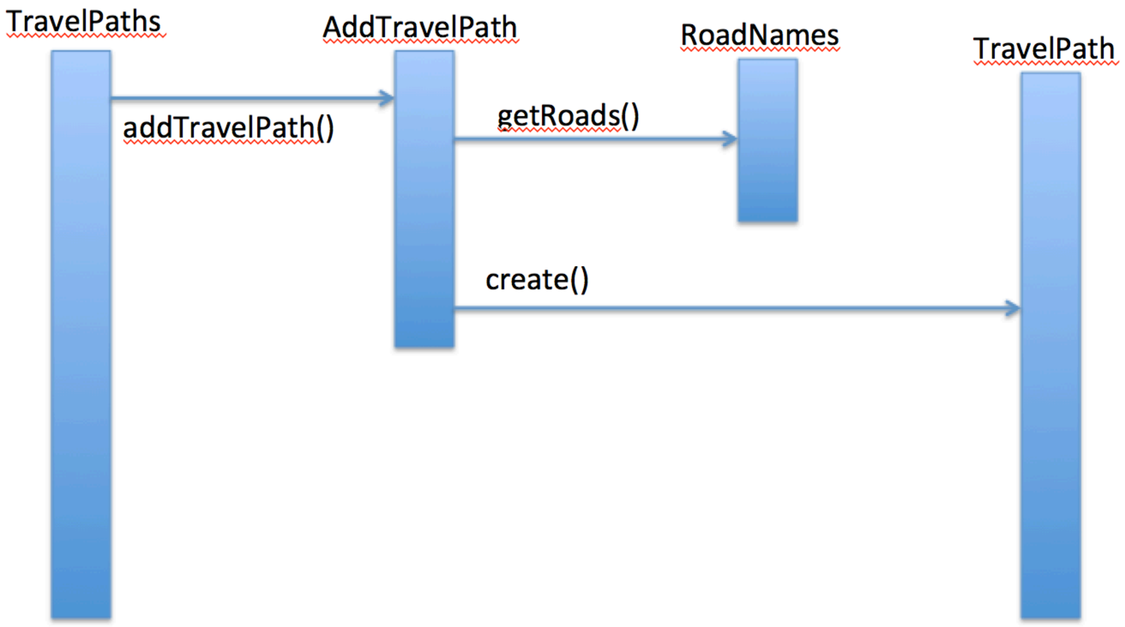
Interaction (Sequence) Diagram – Use Case 3: Update Data by User



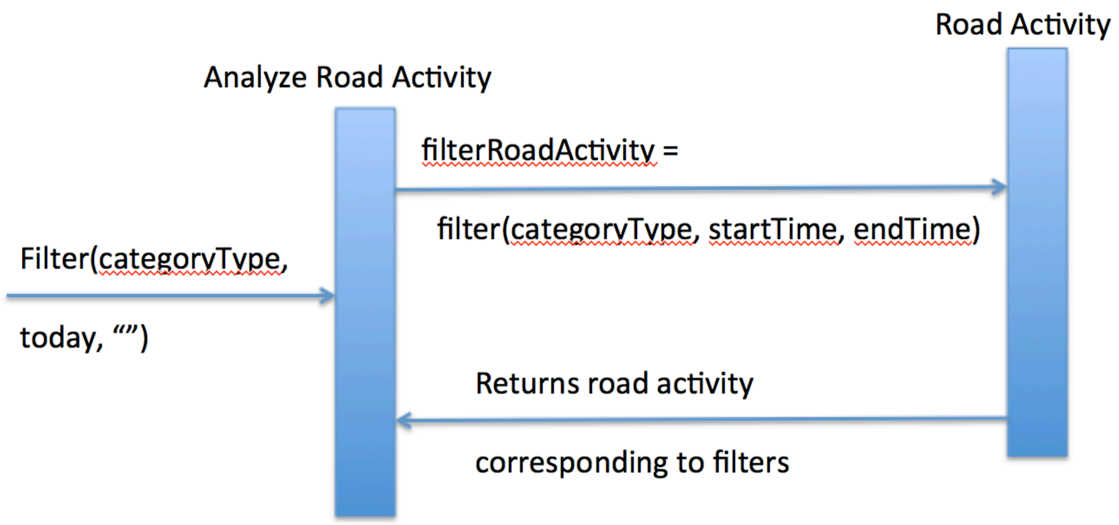
Interaction (Sequence) Diagram – Use Case 4: Analyze Road Activity



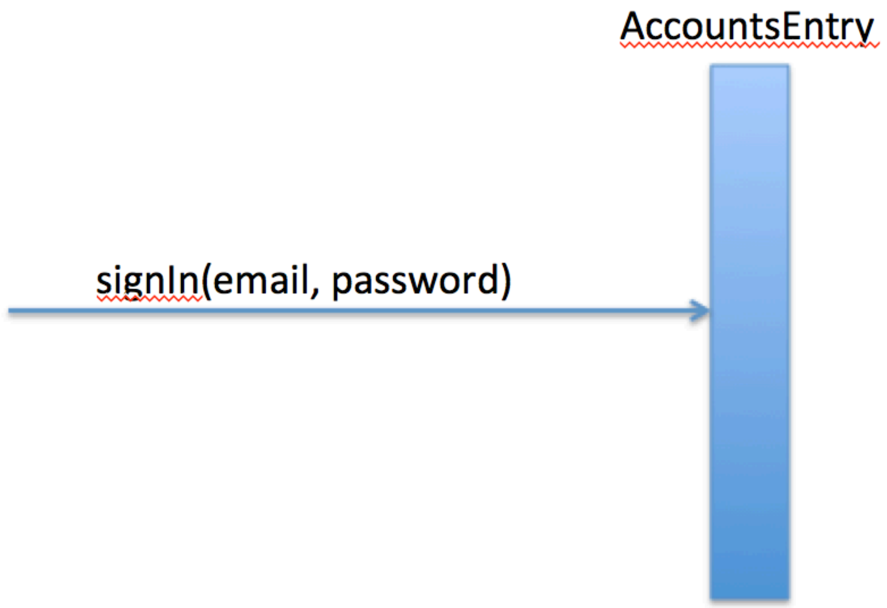
Interaction (Sequence) Diagram – Use Case 5: Enter Roads Travelled



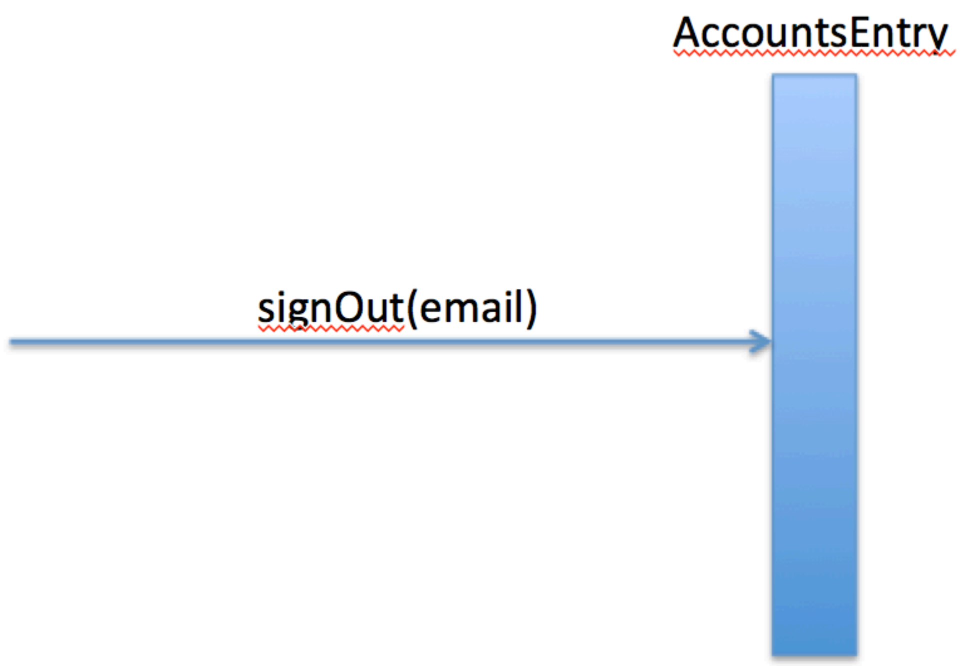
Interaction (Sequence) Diagram – Use Case 9: Current Road Activities



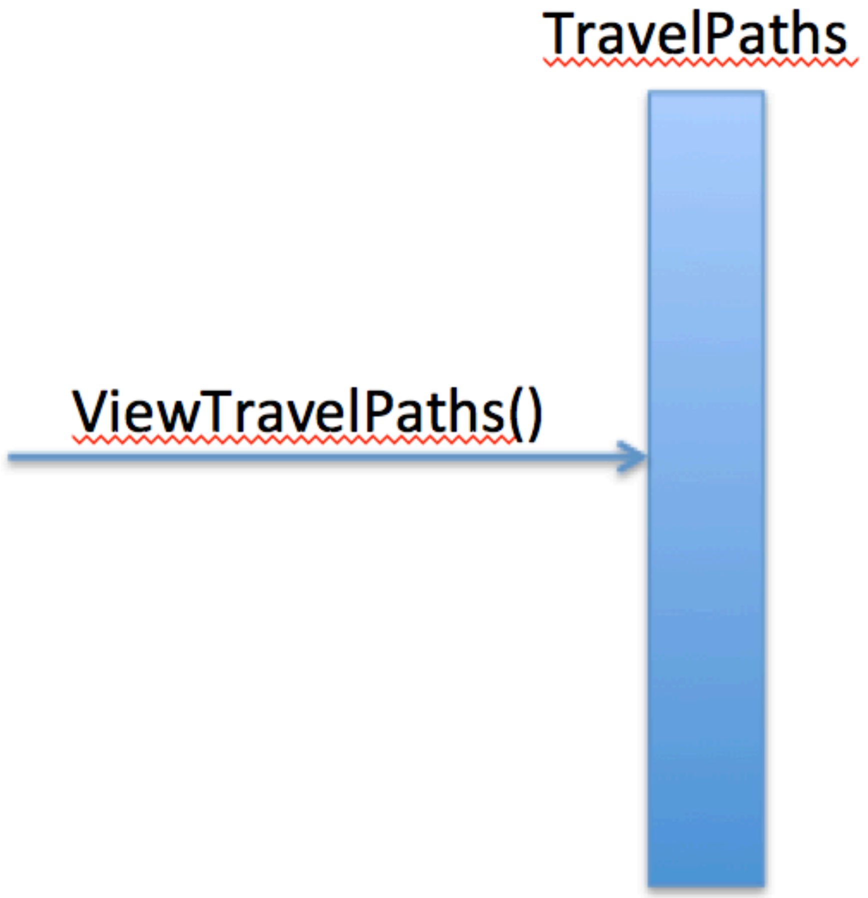
Interaction (Sequence) Diagram – Use Case 10: Sign In



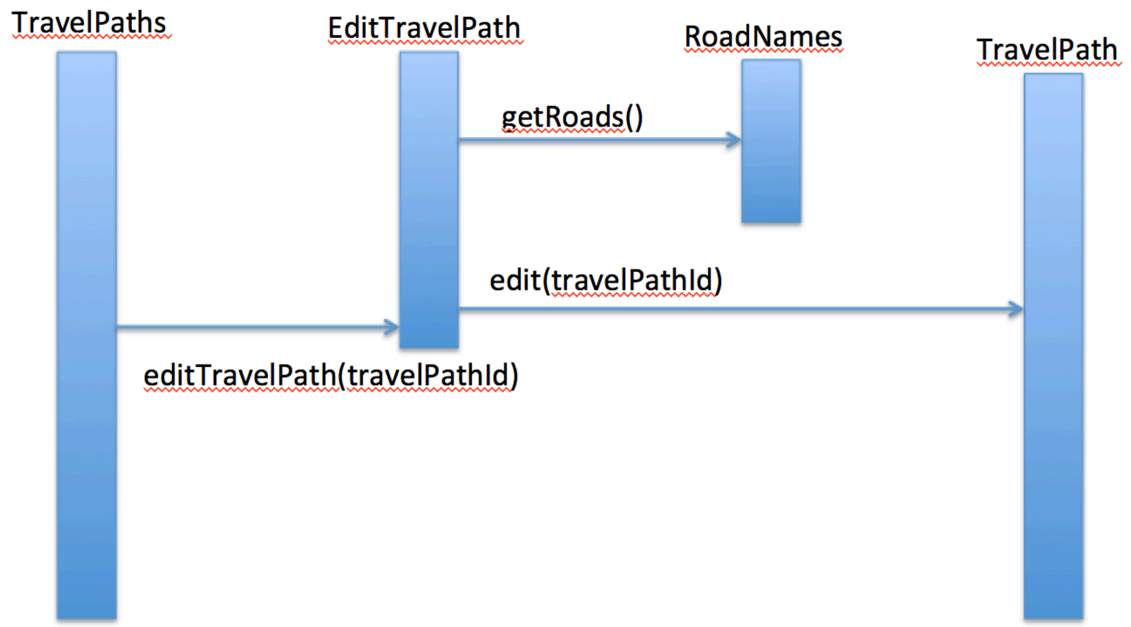
Interaction (Sequence) Diagram – Use Case 11: Sign Out



Interaction (Sequence) Diagram – Use Case 14: View All Travel Paths



Interaction (Sequence) Diagram – Use Case 15: Edit Roads Travelled


TravelPaths
EditTravelPath
RoadNames
TravelPath
getRoads()
edit(travelPathId)
editTravelPath(travelPathId)