José Manuel Ossorio
Marco Antonio Pérez
Juan David Aguirre

Engineering method

Phase 1: Identifying the problem:

- Description of the problem context:
  - There are some people that would like to share their vehicle to help the environment. Slowly over time the usage of cars has grown entailing a bigger and constant impact on the environment produced by the gases as a byproduct of fuel combustion.
- Description of the problem:
  - The usage of cars by the population should be lowered which would translate into a reduction of gas emissions by cars.
- Functional Requirements:
  - The software must:
    - R1: show all close by objects related to the solution (be it cars or stations)
    - R2: zoom in and out on the map
    - R3: let the user pick a location
    - R4: inform the user of the closest available vehicle to such location
    - R5: inform the user of the distance between the selected location and the closest available vehicle
- Non-functional requirements:
  - The software must:
    - Easily implemented
    - Easily developed
    - Relatively low cost

Phase 2: Compilation of necessary information:
Sources:

- https://www.transportenvironment.org/sites/te/files/publications/Does-sharing-cars-really-reduce-car-use-June%202017.pdf
- https://www.itf-oecd.org/shared-mobility-innovation-liveable-cities

While the document does not focus primarily on the environmental side of the situation at hand, the evidence indicates towards the idea of ride-shares as a means to reduce the ever growing car usage. The reduction of car usage would translate into a reduction of car-related gas emissions and overall more liveable cities.

Information regarding the solution itself:
Sources:

- http://www.independent-software.com/gmap-net-beginners-tutorial-adding-polygons-and-routes-to-your-map-updated-for-vs-2015-and-gmap-net-1-7.html

We researched how to use the GMap.NET control in order to solve the problem through the use of this tool. We did a demo project which had some markers on the map, some routes and also a polygon. All this to get used to GMap.NET

Key elements in the solution of the problem

- Marker: a marker on the map
- Route: a line between 2 or more markers
- Distance: GMap.NET can calculate the distance of a route. This is key for our problem because we need to find the closest vehicle to a position
- Latitude and Longitude: these are a way to locate points on the map, which would be useful to mark the locations of the vehicles

Phase 3: Creative solutions

We used the brainstorming method to come up with these ideas
All solutions will be implemented in software form, all of these will allow:

1. the user to visualize a map of his or her surroundings which displays all of the vehicles close by, the service wouldn't be charged.
2. the user to visualize pre-established “car stops” where people willing to share their cars can pick others up.
3. users to share their car for a minimum fee.
4. the user to visualize checkpoints where authorities would enforce a country wide “car-share” policy in which people should always share their car so long there’s less than two people in it

Phase 4: Transition to preliminary design

Options 2 and 4 are immediately discarded as the former requires a large economical inversion and the latter would come into trouble with the desires of people who do not feel comfortable with having strangers in their car, furthermore, the creation of laws can take a considerable amount of time.

1. Application without fees:
   For the user input:
   a. Have a field for the user to introduce the latitude and longitude of his current position
   b. Have a click listener for the user to click anywhere on the map and get that position’s latitude and longitude
   c. Have fields for the user to introduce and address
   d. Implement a MouseHover method that would be indicating at all times what the closest vehicle to the current mouse’s position on the mouse is

   For finding the closest vehicle:
   e. Go through all the markers (vehicles’ position) comparing distances and finding the shortest one
   f. Make a graph with the user’s location and all the other vehicles

g. Make a reasonable area around the user's input location so that we don't have to go through all the possible markers, since some of them will clearly be too far and wouldn't need to be considered when finding the closest one

2. Application which uses fees:

This application would be considerably similar to the other one proposed with the only difference being that it should implement the following:

a. A payment system
b. Strong security to manage the payment information

Phase 5: Evaluating and selecting the best solution

For the sake of better judgement into the evaluation of solutions, the following rubric was designed:

1. Development difficulty
    a. 1 - High difficulty
    b. 2 - Medium difficulty
    c. 3 - Low difficulty
2. Implementation difficulty
    a. 1 - High difficulty
    b. 2 - Medium difficulty
    c. 3 - Low difficulty
3. Governmental implications
    a. 1 - High implications [Such as regulations which could lead to illegality]
    b. 2 - Medium implications [Such as regulations as to how many people per car or traffic]
    c. 3 - Scarce implications [Involvement of the government in marketing situations]
    d. 4 - No implications

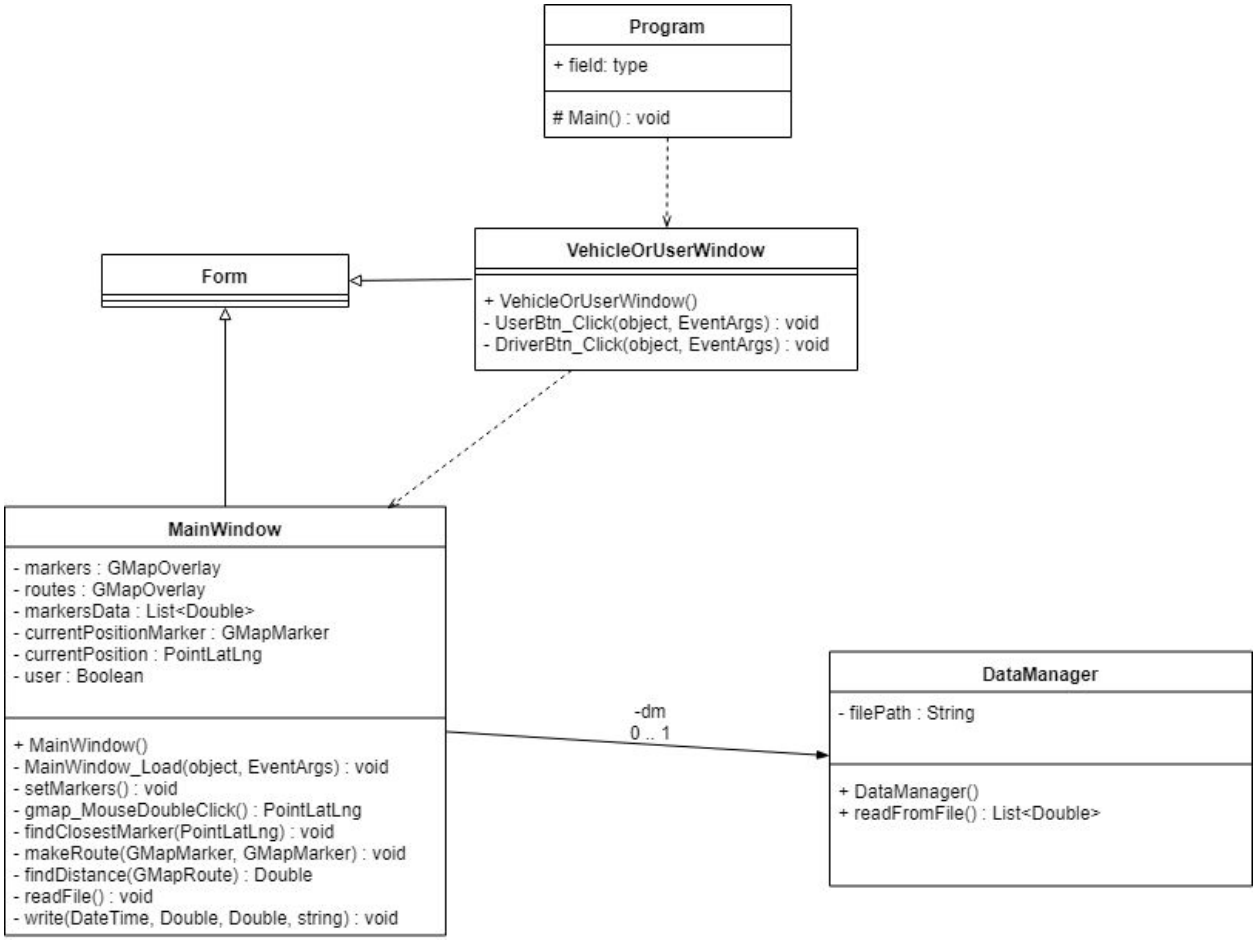
Evaluation:

Solution	**1**	**2**	**3**	**Total**
1	2	2	3	**7**
2	2	2	1	**5**

The results direct us towards the design of an application which does not require a fee.

Phase 6: Preparation of reports and specifications

Class Diagram:



Object Diagram: