Middle East Technical University

Department of Computer Engineering

Computer Engineering Design I
Fall 2011

Software Requirement Specifications Report
for PATİKA project

Strider of Modern World

Levent Oral
Gözde Özcel
Bahar Şevket
Pınar Yılmazer

Table Of Contents

1. Introduction.....5

1.1 Problem Definition.....5

1.2 Purpose.....5

1.3 Scope.....5

1.4 User and Literature Survey.....6

1.5 Definitions and Abbreviations 7

1.6 References 7

1.7 Overview.....7

2.Overall Description.....8

2.1 Product Perspective.....8

2.2 Product Functionalities.....8

2.3 Constraints, Assumptions and Dependencies.....10

3.1 Interface Requirements.....11

3.1.1 User Interfaces 11

3.1.1.1 Main Screen.....11

3.1.1.2 Vehicle Screen.....11

3.1.1.3 Add Destination Points Screen.....12

3.1.1.4 Map Operations 12

3.1.2 Hardware Interfaces.....12

3.1.3 Software Interfaces.....13

3.2 Functional Requirements.....13

3.2.1 Vehicle Operations.....13

3.2.1.1 Background Information.....13

3.2.1.1 Simulus/Response Sequences 14

3.2.1.2.1 Diagram 14

3.2.1.2.2 Description.....15

3.2.1.2.3 Normal Flow Of Events.....15

3.2.1.2.4 Alternative Event Flows.....16

3.2.1.3 Functional Requirements... 17

3.2.2 Locate User.....17

3.2.2.1 Background Information.....17

3.2.2.2. Stimulus/Response Sequences.....18
3.2.2.2.1 Diagrams.....18
3.2.2.2.2 Description 18
3.2.2.2.3 Normal flow of Events 18
3.2.2.2.4 Alternative Event Flows.....18
3.2.2.3 Functional Requirements.....20
3.2.3 Visualise Map.....20
3.2.3.1 Background Information.....20
3.2.3.2. Stimulus/Response Sequences.....20
3.2.3.2.1 Diagrams.....20
3.2.3.2.2 Description 21
3.2.3.2.3 Normal flow of Events 21
3.2.3.2.4 Alternative Event Flows.....21
3.2.4 Point Adding 21
3.2.4.1 Background Information.....21
3.2.4.2. Stimulus/Response Sequences.....22
3.2.4.2.1 Diagrams.....22
3.2.4.2.2 Description 22
3.2.4.2.3 Normal flow of Events 23
3.2.4.2.4 Alternative Event Flows.....24
3.2.5 Show Path 26
3.2.5.1 Background Information.....26
3.2.5.2. Stimulus/Response Sequences.....27
3.2.5.2.1 Diagrams.....27
3.2.5.2.2 Description 27
3.2.5.2.3 Normal flow of Events 27
3.2.5.2.4 Alternative Event Flows.....28
3.3 Non-Functional Requirements.....28
3.3.1 Performance Requirements.....28
3.3.2 Design Constraints.....29
3.2.3 Simplicity.....29
3.2.4Accuracy.....29

4. Data Model and Description 30
4.1 Data Description..... 30
4.1.1 Data Objects 30
4.1.2 Relationships 30
4.1.3 Complete Data Model.....32
4.1.4 Data Dictionary 32
4.1.4.1 Map.....32
4.1.4.2 Point.....32
4.1.4.3 Vehicle 33
5. Behavioral Model and Description 33
5.1 Description of Software Behaviour 33
5.2 State Transition Diagrams.....35
6. Planning 36
6.1 Team Structure.....36
6.3 Process Model 37
7. Conclusion 38

1. Introduction

This Software Requirements Specification report is prepared by “SMW” group members to determine 3D Map Visualization and Path Finding project requirements. It is designed to serve readers to understand the concepts of the project and to use this tool easily and effectively. An overview of the problem and the product are described. Specific requirements and, data model and descriptions are also introduced. It is aimed to serve readers to understand the concepts of the project and to use this tool easily and effectively.

1.1 Problem Definition

Military logistics is a process of planning and carrying out the movement and maintenance of military forces. Therefore, fuel delivery for military vehicles has an important place in military logistics and it must be handled in most efficient way. Today, fuel delivery systems are planned with the use of different type of maps and software applications. While vector maps, which are most widely used maps, contains features like coordinates, major road networks, railroad networks and international boundaries, they do contain the knowledge of height of a point. However, military vehicles are capable moving on not only human made roads, but also different types of terrains. As a result, while calculating shortest distance and shortest trip time for fuel delivery purposes, the topographic map, maps showing height of a particular area, must be used. Moreover, the vehicles capabilities such as moving on different terrain types, moving on a slope and fuel amount should be considered. Although there are many applications offering shortest path finding solution in the market, there is no such a platform, which satisfies all the needs of users mentioned above. Thus, in this project these issues will be addressed.

1.2 Purpose

This Software Requirement System is written in order to provide a complete description of all the functions and specifications of 3D Map Visualization and Path Finding project. It introduces basic functionalities, use cases, constraints and dependencies of the project. It also gives information about product, planning and data model. Written descriptions and types of modeling diagrams is used to demonstrate the high level structure of the program. Because this project is first intended to be used for military purposes, our audience is military industry workers and prospective software developers associated with this project. Therefore, another purpose of this document is to guide all audience in the best way.

1.3 Scope

Overall description, specific requirements, behavior model and general information of the project is described in this document.
The software to be produced is a 3D map visualization and part finding for fuel delivery system with different terrains and characteristic features of vehicles for Android. It will generate the shortest path between stationary and moving points. It will support distinct characteristics of different military vehicles and different terrains in order to be applicable to

real world.
In addition to these, the product will visualize generated paths, air routes, and virtual areas on air on 3D maps and also have a guiding feature in order to be more user friendly. Because it will be capable of mobile devices using Android Operating System, it will not be CPU intensive.

1.4 User and Literature Survey

Users of the Project:

This project is intended to be used for military purposes since our sponsor is ASELSAN. It will be integrated to a large variety of devices used in military. It aims to fuel supply to any point in the map by using shortest path approach. Also, it will be compatible with Android which makes it useable for mobile devices.Therefore, users of this project will be military industry workers and prospective software developers associated with this project.

Market Research

There are many technologies that refers to the "path finding" problem in the market which generally refers to the plotting, by a computer application, of the shortest route between two points. It has grown in importance as games and their environments have become more complex. There are two commonly used algorithm for this purpose which are Dijkstra and A*. A* is a variant of Dijkstra's algorithm commonly used in games. Instead of looking at the distance from the start node, A* chooses nodes based on the estimated distance from the start to the finish [1]. It is extremely configurable to the particular type of game and map and efficient in path finding against most data sets, especially when there're chasms, rocks or other obstacles on the way[2]. That's why most games and other applications which are developed in Android platform use A* path finding algorithm. We will also take it as a base while developing path finding algorithm for "patika".

Most of the applications use waypoint graphs for path finding. However, there are several disadvantages of using waypoint graphs in our project. In order to compensate for these disadvantages, we will be using navigation mesh approach. In the market, there are several applications using navigation mesh for path finding but they did not use OpenGL and dynamically changing Dted maps. Also, in applications developed for military purposes, most of the algorithms use vector maps. However, we will be implementing path finding algorithm without vector maps.
Besides game technologies and GPRS applications, this algorithm is not implemented for Android sufficiently. However, there is a similar application, The Military Unit Path Finding[6] which solves the problem of finding a path from a starting point to a destination where a military unit has to move, or be moved, safely whilst avoiding threats and obstacles and minimizing costs in a digital representation of the real terrain. The two projects seem similar in many aspects. However, in the MUPF, images which are from satellite sources, aerial photographs, Google Earth are used by converting provided image to a 2D grid. It was also developed for PC, not Android platform.

To sum up; in the market there are many applications that use path finding on 3D

system but they developed mostly for the purpose of game technologies. Ones that are developed by military purposes use mostly vector maps for visualization, they do not support dynamically changing DTED maps and they are not suitable for Android platform. Therefore, there is not any available solution for our problem in the market.

1.5 Definitions and Abbreviations

SMW : Striders of the Modern World
SRS: Software Requirements Specification
PC : Personal Computer
CPU : Central Processing Unit
DTED : Digital Terrain Elevation Data
WIFI : Wireless Fidelity
GPS : Global Positioning System
GPRS : General packet radio service
DB : Data Base
ERD : Entity Relationship Diagrams
MUPF : Military Unit Path Finding
IEEE STD 830-1998: IEEE Recommended Practice for Software Requirements Specifications

1.6 References

[1] http://en.wikipedia.org/wiki/Pathfinding
[2] http://forum.9kgames.com/default.aspx?g=posts&t=128
[3] http://researchspace.csir.co.za/dspace/bitstream/10204/4200/1/Leenen_2010.pdf
[4]https://cow.ceng.metu.edu.tr/Courses/download_courseFile.php?id=3574
[5] IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications
[6] http://www.gliffy.com
[7] http://developer.android.com/guide/topics/graphics/opengl.html

1.7 Overview

This document has six main sections. First section is introduction part which gives an overview of the entire software requirement system. Second section is about overall description part which tells the general factors that affect the product and its requirements. Details are avoided and understanding of the overall project is aimed. Third section contains specific requirements which explain all the software requirements to a level of detail in order to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. Modules and interfaces are also explained in detail in this section. Fourth section is about data model and description which gives information domain for the software. It contains information about data models, relations with each other and data flow. Fifth section explains behavioral model and description using state transitions and last section is about planning the team structure, estimation and process model. Planning of the project, distribution of group members to specific parts of the project

is mentioned in this section. There is also a conclusion section which summarizes all details in the report[4].

2. Overall Description

This section will provide background information about software requirements.Information will be given in terms of product perspective, product functions and contraints, assumptions and dependecies.

2.1 Product Perspective

Patika Application will be an application that run in Android Platforms to find a shortest path between multiple points on a map which is visualized using height maps.Applicaion will be used for military logistic fuel-carry purposes.

Patika Application will be an independent application and it has only one type of user support so application does not contain any functionality for communication between multiple users.Because of application will be an independent system it only requires Android Platform to start and use application.There is also a need for WiFi/GPS support for usage of application because of functionality supports.Application will have one type of interface for one type user.The interface application will contain a map which is visualised using height maps espeacially DTED2 data.Visualisation part will be done by using OpenGL ES API which is Open GL for Android device or Android NDK for implementation of OpenGL[7].Users will see their locations on map via GPS/WiFi/UserDefined methods.User can add points on map to find shortest path between points which is time dependent or fuel(economic) dependent.Path will be calculated and shown to users in terms of dependency choice of user and vehicle type of users.To enable support of multi-type-vehicle usage, users will can manipulate vehicles and select specific type of vehicle for usage in calculations.Interface of application will have simple screen for vehicle operations for more simpler usage by users.

2.2 Product Functionalities

There are five major functionalities of application :

a) Locating User :

The program will locate user on map as a point on map by using GPS, WiFi or User Defined methods.Firstly, application will try to get location of user GPS of WiFi by requesting user to open one of them and then by using data that coming from one these

channels.If these methods will both fail in some way or user thinks there is an error on location, users will be asked by program or will change voluntarily location of themselves by entering their latitude, longtitude and altitude.

b) Visualising Map and Gesture Actions on Map :

Application will visualise map for demonstrations and actions to users by using DTED2 data.User can zoom in, zoom out and change visible position of map by gesture and touch actions

c) Vehicle Manipulations :

Application will enable users to add, edit, remove and select vehicle that is used in program.There is predefined sample vehicles in application so that user will not have to implement a vehicle to use application.After select a vehicle, program will automatically save choice of user for later usages.

d) Point Adding :

User will be able to add points and change them on map.Moreover, user can bookmark a point for later usages.There will be a recent points option for fast and active usage of application.Users will be able to choose points by using bookmarks, recents or newly defined points for current usage.

e) Path Finding and Demonstrating :

Users will choose dependency of shortest path finding.In other words, users will demand time-dependent shortest path or fuel(economic)-dependent shortest path on points that is added previously.Application will show path in order.After start to movement application will change current position and target position of user.If there is a fail updating user location, program will wait from user to update current location only in terms of completing current action,i.e reach next position with one tick.

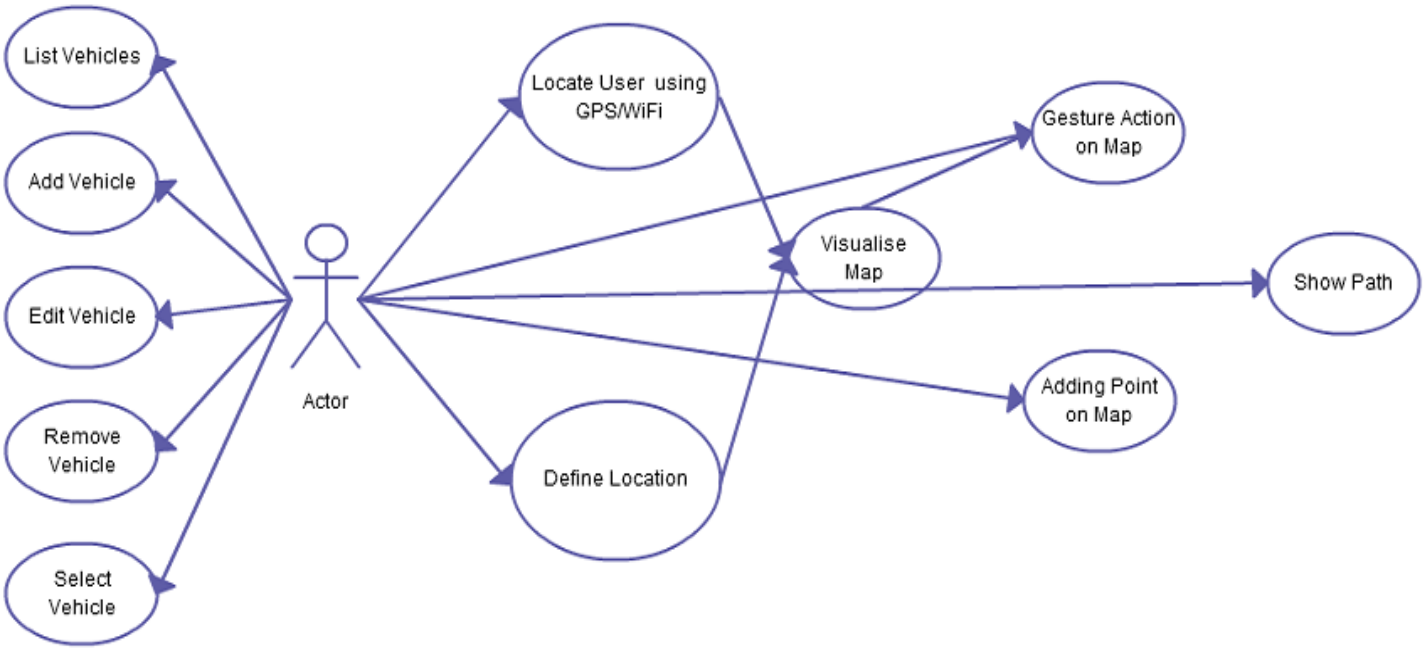


Figure 1 – Use Case Diagram

2.3 Constraints, Assumptions and Dependencies

- The system will dynamically track the user's position on the earth, therefore it will need position data either coming from Internet or GPS.

- When user move through map and want to see the regions outside his/her viewing region, that is limited with screen size of the device, the software will generate the new parts of the map by reading map data from database. This visualization of new parts of the map should not take more than 20 seconds. If data about the wanted region is not available on the device, the software will search in the internet. If the Internet connection is not available, user should be notified in less than 3 seconds.

- The software must be multi-threaded, because there will be processes that will be indepented from each other and also when the software makes neccessary calculations user should be able to interract with the program.

- The software will run on Android OS, therefore the target devices will be mobile devices. As a result, the graphical user interface should be designed with taking sizes of mobile devices into consideration.

3.1 Interface Requirements

3.1.1 User Interfaces

In this part of the report main GUI elements of our project will be explained, namely main screen, vehicle screen, add destination points screen and map operations. Morover, since our project will run on Android OS, we have to consider different sizes of different devices that use Android OS. Therefore, the locations and sizes of the screens and buttons described below can be designed differently according to different devices.

3.1.1.1 Main Screen

The map and the current location of the user will be displayed in the main screen. The map which will displayed will be chosen by the software independently from user, according to the current location of the user. The current location will be determined according to the data coming from GPS or Wifi. Then the corresponding map will be searched from local database and if it is not found then it will searched from Internet. The generated shorthest path will also be displayed on the main screen. Moreover, according to the results of map operations such as zoom in/out the map will be updated.

3.1.1.2 Vehicle Screen

The user can open the vehicle screen by tabbing the vehicle menu button which will be located on main screen. In this screen the predefined vehicles will be displayed and the user will be able to perform operations such that “Add new vehicle”, “Delete a vehicle”, “Edit a vehicle” and “Select a vehicle”. The vehicles will be displayed with corresponding name and an image in the vehicle screen. There will be add button, when user tabs it a new window will be opened. The user will be able to add features to a new vehicle in that window. Some of the possible features are name of the vehicle, maximum speed, fuel capacity, the maximum incline that vehicle can climb, etc. When user tabs on a vehicle in the main vehicle screen, a new window will be opened that will display the detailed information about the vehicle. On that window there will be delete , select and edit buttons. They will enable user to perform corresponding actions.

3.1.1.3 Add Destination Points Screen

This screen will be opened when user tabs the corresponding button in the main screen. However, the map will still be visible, this screen will only be shown on the part of the whole screen. The users can add destination points by four different ways, namely by entering the coordinates, by selecting a point from recently added points, by selecting a point by predefined points and by tabbing on the map. When user selects a point by tabbing on the map, a info box will be opened on the map displaying the coordinates of the corresponding point and if user clicks on the add button which will be displayed on the info box, the point will be added. Our application will support adding multiple points, therefore user must tab to a button like “generate shortest path” in order to program can call the shortest path algorithm.

3.1.1.4 Map Operations

The map operations and gestures can be performed on the main screen in which map is displayed. The user will be able to zoom out/in and to move right/left/up/down on the map. In order to zoom out/in by performing zoom gestures defined for touch screens. The zoom gestures are performed simply by touching to the screen with two fingers and moving them toward or away from each other. Moreover, the zoom operations also can be performed by tabbing the corresponding buttons which will appear in the screen. In addition, in order to move to different directions on the map, user should touch on the screen with one finger and move his/her finger to corresponding directions.

3.1.2 Hardware Interfaces

The software will run on devices with Android OS. Therefore, the device can either be a mobile phone or a tablet PC. In order to obtain location information, the software will need a GPS or Wifi support. Moreover, the 3D maps will be displayed and the shortest path algorithm will be implemented, therefore the hardware should have a reasonable CPU speed, in order to perform this operations in reasonable time.

3.1.3 Software Interfaces

As mentioned before, the software will run on Android OS, therefore in development stage we will be using Android SDK. Furthermore, the software will require database in order to store vehicle features and predefined destination points, as a result we will use MySQL as database management system and SQL will be used for communicating with database. For communication purposes the software will be using GPS and wireless. TCP/IP protocol will be used for wireless communication.

3.2 Functional Requirements

3.2.1 Vehicle Operations

3.2.1.1 Background Information

Vehicle Operations Interface enables user to perform basic vehicle based operations which are shortly adding, deleting, updating and listing operations.

Add Vehicle

User can be able to add a new vehicle to the system either choosing it from the sample vehicle list or creating it by defining necessary features. Sample vehicle list keeps the basic predefined vehicle instances, which are appropriate to operate on most common landforms. If user cannot find proper vehicle from this list, he/she can create a new vehicle instance by constructing its fields. Main fields are name, maximum and minimum slope it can climb, capability of moving in the water and night vision, velocity and fuel information. If user can not fill any of these fields, he/she is notified.

Update Vehicle

User can be able to update vehicle by choosing it from the list it. Updated fields are determined according to variable landforms. If user leaves any necessary fields blank after updating, he/she is notified.

Delete Vehicle

If there is unused vehicle instance in the list, user is free to remove it from the system.

Select Vehicle

User can be able to select vehicle from the list in order to achieve adding, updating and deleting methods. Moreover, he/she can see picture and details of vehicle instance with select operation.

3.2.1.1 Simulus/Response Sequences

3.2.1.2.1 Digram

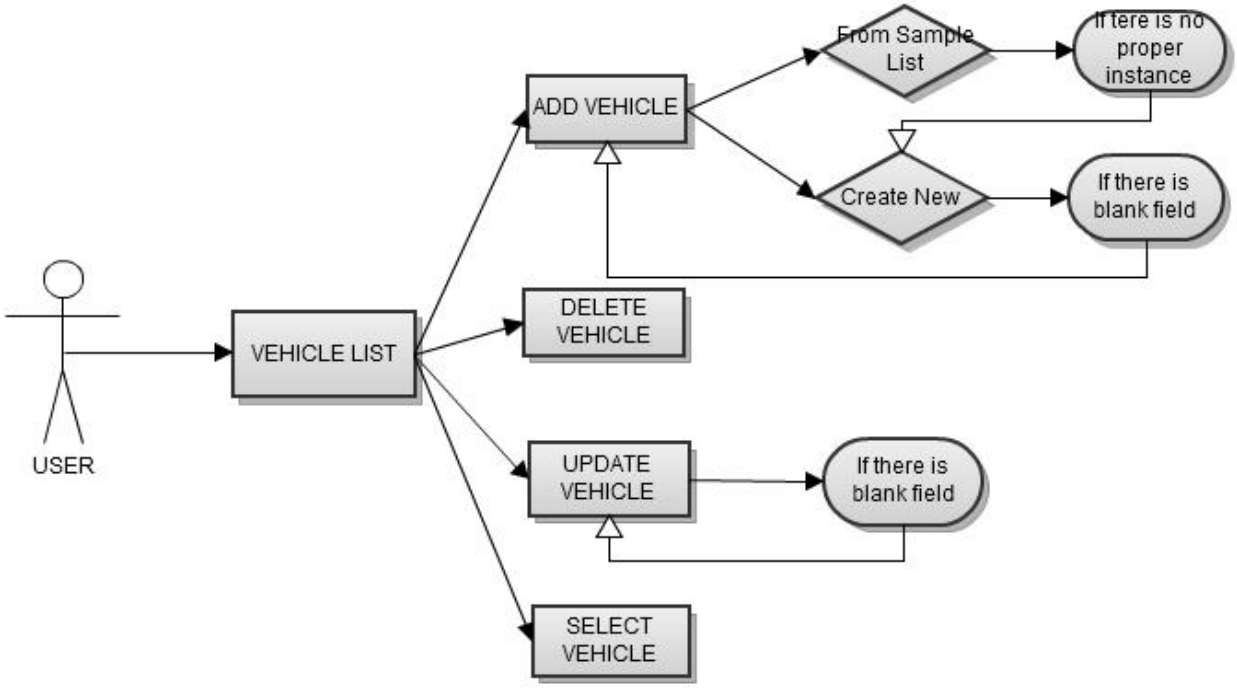


Figure 2 – Use Case Diagram for vehicle Operations

3.2.1.2.2 Description

Aim	This feature is to make basic vehicle operations which are adding, deleting, updating and deleting vehicle instance.
Precondition	No precondition is defined.
Trigger	User wants to manipulate current vehicle list by applying several operations.

3.2.1.2.3 Normal Flow Of Events

Add Vehicle Flow 1

1.User opens vehicle operations interface.

2.User controls sample vehicle list in order to find proper vehicle instance.

3.User selects vehicle from sample vehicle list.

4.User clicks Add button from the detail part of the selected vehicle.

5.Vehicle is successfully added to main vehicle list.

6.User is redirected to vehicle operations interface.

Add Vehicle Flow 2

1.User opens vehicle operations interface.

2.User controls sample vehicle list in order to find proper vehicle instance.

3.User clicks New button and new vehicle form is opened.

4.User fills the necessary fields in the form and submits it.

5.Vehicle is successfully added to main vehicle list.

6.User is redirected to vehicle operations interface.

Update Vehicle Flow 1

1.User opens vehicle operations interface.

2.User selects vehicle which will be updated from main vehicle list.

3.User clicks Update button from the detail part of the selected vehicle

and update form is opened.

4.User updates necessary fields in the form and submits it.

5.Vehicle is successfully updated.

6.User is redirected to vehicle operations interface.

Delete Vehicle Flow 1

1.User opens vehicle operations interface.

2.User selects vehicle which will be deleted from main vehicle list.

3.User clicks Delete button from the detail part of the selected vehicle and update form is opened.

4.Vehicle is successfully deleted.

5.User is redirected to vehicle operations interface.

Select Vehicle Flow 1

1.User opens vehicle operations interface.

2.User traverses the main vehicle list.

3.User selects vehicle and detail part is appeared.

4.User examines vehicle information.

5.User achieves some operations.

6.User is redirected to vehicle operations interface.

3.2.1.2.4 Alternative Event Flows

Add Vehicle Alternative Flow 1

3.User cannot find proper vehicle.

4.User follows add vehicle flow 2.

Add Vehicle Alternative Flow 2

5.User leaves any of the fields blank.

6.User is notified that fields cannot be empty.

Update Vehicle Alternative Flow 1

5.User leaves any of the fields blank.

6.User is notified that fields cannot be empty.

3.2.1.3 Functional Requirements

REQ 1: The system shall check if there are any blank fields while adding and updating operations.

3.2.2 Locate User

3.2.2.1 Background Information

Locating the user is a key feature for finding the various efficient paths. This feature requires either GPS or Wifi to determine the location of the user online. GPS will be the primary tool, and Wifi will not be used unless a device does not support GPS. If there's no chance of any online connections, this feature will also support manual coordinate entering. To be able to use Wifi to locate the user, we will have to use an underlying Wifi to GPS API. Upon entering the application, user will be prompted to turn on GPS if it's not already enabled or absent. Same will be the case for Wifi. If neither are available, user will be prompted to enter her/his location manually.

3.2.2.2. Stimulus/Response Sequences

3.2.2.2.1 Diagrams

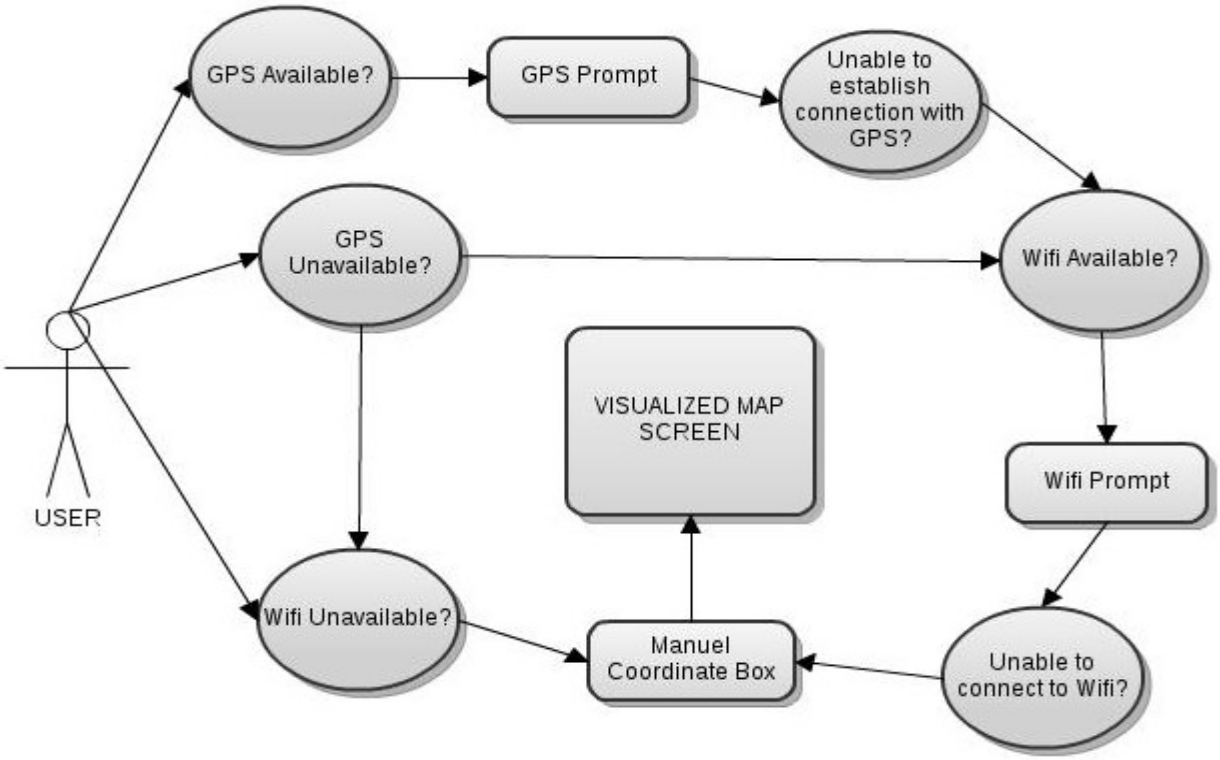


Figure 3: Use Case Diagram for Locate User

3.2.2.2.2 Description

Aim	Aim of this feature is to put an interface to prompt user to turn on online tracking tools to determine the location of the user.
Precondition	No precondition.
Trigger	User wants to locate her/his position on the map.

3.2.2.2.3 NORMAL EVENT FLOWS

NORMAL FLOW OF EVENTS (GPS)

1. User opens the application.
2. GPS Prompt screen appears.
3. User chooses to turn on GPS.
4. GPS is succesfully turned on.
5. User is redirected to the Map Visualization Screen.

NORMAL FLOW OF EVENTS (WIFI)

1. User opens the application.
2. Wifi Prompt screen appears.
3. User chooses to turn on Wifi.
4. Wifi is succesfully turned on.
5. User is redirected to the Map Visualization Screen.

3.2.2.2.4 ALTERNATIVE EVENT FLOWS

ALTERNATIVE EVENT FLOW 1 (GPS)

4. Unable to connect to GPS.
5. Wifi Prompt appears.
6. User chooses to turn on Wifi.
7. Wifi is succesfully turned on.
8. User is redirected to the Map Visualization Screen.

ALTERNATIVE EVENT FLOW 2 (GPS)

2. GPS unavailable.
3. Program checks if Wifi is available.
4. Wifi Prompt screen appears.
5. User chooses to turn on Wifi.
6. Wifi is succesfully turned on.
7. User is redirected to the Map Visualization Screen.

ALTERNATIVE EVENT FLOW 1 (WIFI)

4. Unable to connect to Wifi.
5. Manuel Coordinate Box appears.
6. User enters coordinates.
7. User is redirected to the Map Visualization Screen.

ALTERNATIVE EVENT FLOW 2 (WIFI)

2. Wifi unavailable.
3. Manuel Coordinate Box appears.
4. User enters coordinates.

5. User is redirected to the Map Visualization Screen.

3.2.2.3 FUNCTIONAL REQUIREMENTS

REQ1 : System shall check if GPS is available.

REQ 2: System shall check if connection to GPS is established correctly.

REQ 3 : System shall check if Wifi is available.

REQ 4 : System shall check if connection to Wifi is established correctly.

3.2.3 Visualise Map

3.2.3.1 Background Information

Visualizing the map will be done using DTED2 maps. This feature requires reding of DTED2 data, processing it and then visualizing it using Open GL. Also, the visualized map will be controlled by gesture actions. Thus, one of the important requirements of the feature is Gesture Action support. Basic gestures that will be implemented are up,down,left,right direction touches, zoom in – zoom out touches and choosing map location to go to.

3.2.3.2 Stimulus/ Response Sequences

3.2.3.2.1 Diagrams

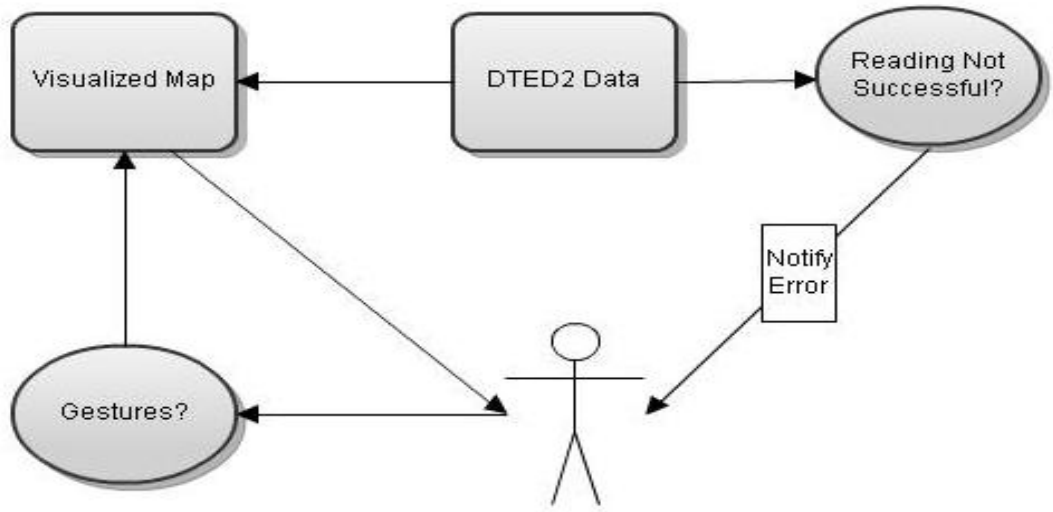


Figure 4: Use Case Diagram for Visualise Map

3.2.3.2.2 Description

Aim	Aim of this feature is to put a graphical interface to better simulate the area and paths for the user.
Precondition	Gesture Actions are supported.
Trigger	Visual information is much more easy to understand and follow.

3.2.3.2.3 NORMAL EVENT FLOW

1. DTED2 files are read.

2. Our application processes the data and then visualizes it.

3. Output of the visualization is shown to user in a screen.

4. User may or may not use gestures, but if s/he does, the screen updates itself.

3.2.3.2.4 ALTERNATIVE EVENT FLOW

1. DTED2 Files are not read succesfully.

2. An error message is sent to the user.

3.2.3.3 FUNCTIONAL REQUIREMENTS

REQ1 : System shall check if DTED2 files are read correctly.

3.2.4 Point Adding

3.2.4.1 Background Information

The user will be able to add points to the map, in order to order to software to generate shortest path between these points. The point adding functions will be performed in the “Add destination points screen”. There will be several ways to add a destination points. User may add a point by tabbing anywhere on the map, by entering coordinates of a point or by selecting predefined point from bookmarks or from recently entered points. Because bookmark options is available, the user should also be able to define a bookmark point in this step.

3.2.4.2 Stimulus/Response Sequences

3.2.4.2.1 Diagram

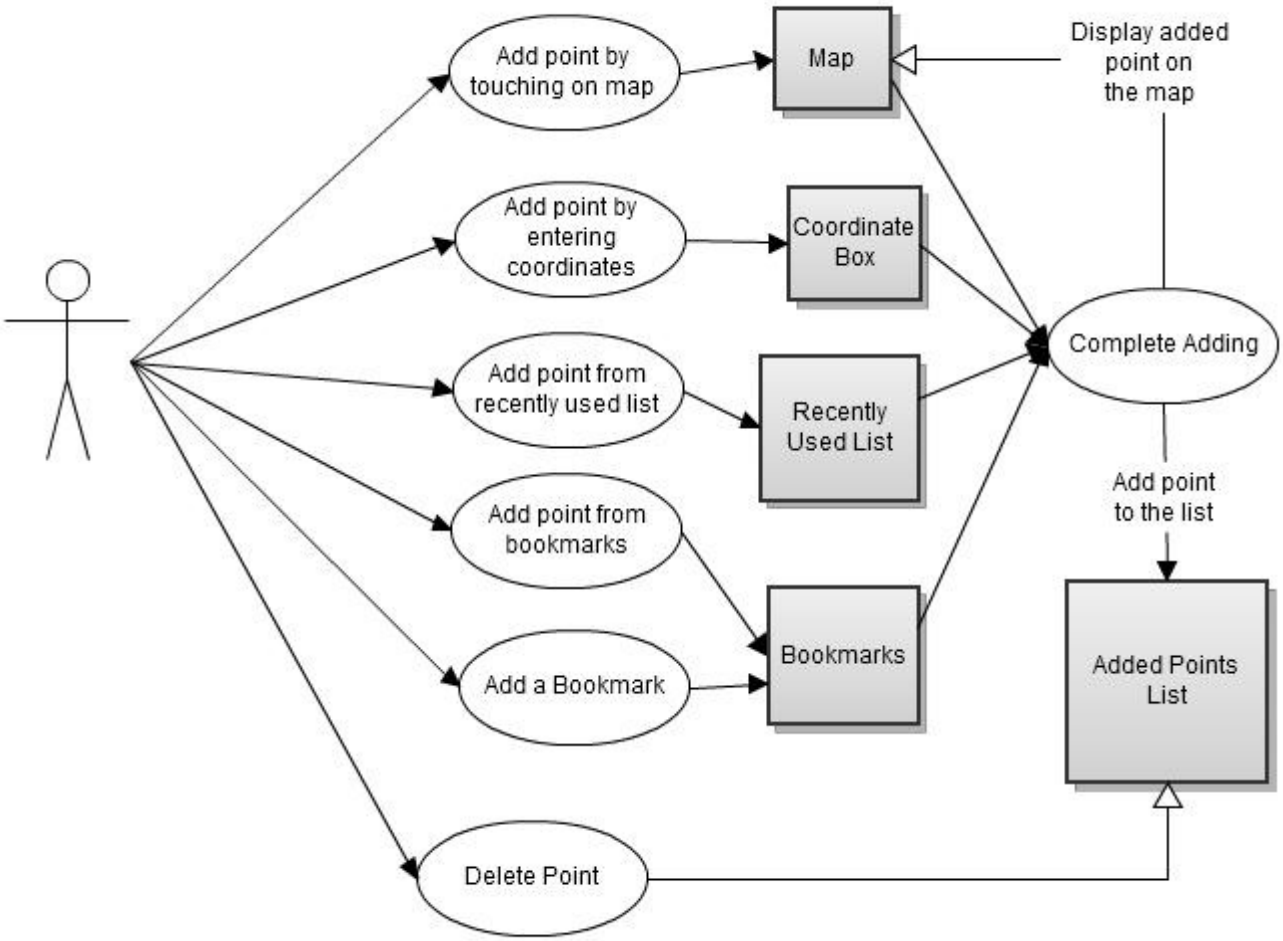


Figure 5: Use Case Diagram for Point Adding

3.2.4.2.2 Description

Aim	To add destination points in order to generate shortest path between user's location and other points
Precondition	Location of the user should be determined and the corresponding map should be available in DB or should be download from Internet.
Trigger	User entering to “Add destination points Screen”

3.2.4.2.3 Normal Flow of Events

3.2.4.2.3.1 Adding Point by Touch on Map

1. User touches on map for at least 2 seconds.
2. A marker for the point and an info box appears on the marker.
3. The coordinates of the point and a button labeled “add” is displayed in info box.
4. User clicks on add button.
5. Point is added to the list and displayed on the map.
6. The user is redirected to the main “Add Destination Points Screen”.

3.2.4.2.3.2 Adding Point by Entering Coordinates

1. User tabs on “Manual Point” button.

2. A small window with text boxes for entering coordinates is opened.
3. User enters coordinates values and tabs on the “finish adding” button.
4. Point is added to the list and displayed on the map.
5. The user is redirected to the main “Add Destination Points Screen”.

3.2.4.2.3.3 Adding Point from Recently Used Points

1. User tabs on “Recent Points” button.

2. A list displaying the recently used points is opened, displaying the coordinates of the points and if point has a bookmark also displaying the bookmark.

3. User chooses a point from list by tabbing on it.

4. A tick appears next to the item, therefore user choose multiple points.

5. User tabs “add point” button.

6. Point or points are added to the list and displayed on the map.

7. The user is redirected to the main “Add Destination Points Screen”.

3.2.4.2.3.4 Adding Point from Bookmarks

1. User tabs on “Bookmarks” button.

2. A list displaying the bookmark used points is opened, also displaying the coordinates of the points.

3. User chooses a point from list by tabbing on it.

4. A tick appears next to the item, therefore user choose multiple points.

5. User tabs “add point” button.

6. Point or points are added to the list and displayed on the map.

7. The user is redirected to the main “Add Destination Points Screen”.

3.2.3.2.3.5 Add Bookmark

1. User tabs on “Bookmarks” button.

2. Bookmark menu is opened.

3. User tabs on “Add New Bookmark” button.

4. A small window with text boxes for entering coordinates and tag of bookmark is opened.

5. User tabs on “Complete Adding” button.

6. Point or points are added to the list and displayed on the map.

7**.** The user is redirected to the main “Add Destination Points Screen”.

3.2.3.2.3.6 Deleting Point

1. User selects a point on the added points list by tabbing on it.

2. A check sign appears next to the item, so user may choose multiple points.

3. Delete button appears when there is at least one selected point.

4. User tabs delete button.

5. The deleted points are removed from list and erased from map.

3.2.4.2.4 Alternative Flow of Events

3.2.4.2.4.1 Adding Point by Touch on Map

3.2.4.2.4.1.1 Alternative Flow 1

1. User touch on the map less than 2 seconds.

2. Nothing appears on the map.

3.2.4.2.4.1.2 Alternative Flow 2

4. User does not tab on add button instead of tab on somewhere else.

5. The marker and the info box dissapears.

6. The user is redirected to the main “Add Destination Points Screen”.

3.2.4.2.4.2 Adding Point by Entering Coordinates

3.2.4.2.4.2.1 Alternative Flow 1

3. User enters incorrect or incomplete coordinate values and tabs on “finish adding” button.

4. A message window is showed to the user, informing him/her that the coordinates values are incorrect.

5. The user is redirected to the main “Add Destination Points Screen”.

3.2.4.2.4.3 Adding Point from Recently Used Points

3.2.4.2.4.3.1 Alternative Flow 1

2. There is no point in the recently used points list.

3. The message box will be displayed, informing user that there is no recent points.

4. The user is redirected to the main “Add Destination Points Screen”.

3.2.4.2.4.3.2 Alternative Flow 2

5. User tabs on “Add point” button without selecting any point.

6. A message box is displayed informing user that he/she did not select any point.

7. The user is redirected to adding point from recently used points screen.

3.2.4.2.4.3.3 Alternative Flow 3

3. User clicks on cancel button at any time.

4. No points is added and user is redirected to the main “Add Destination Points Screen”.

3.2.4.2.4.4 Adding Point from Bookmarks

3.2.4.2.4.4.1 Alternative Flow 1

2. There is no point in the bookmark list.

3. The message box will be displayed, informing user that there is no recent points.

3.2.4.2.4.4.2 Alternative Flow 2

5. User tabs on “Add point” button without selecting any point.

6. A message box is displayed informing user that he/she did not select any point.

7. The user is redirected to adding point from bookmarks screen

3.2.4.2.4.3.3 Alternative Flow 3

3. User clicks on cancel button at any time.

4. No points is added and user is redirected to the main “Add Destination Points Screen”.

3.2.3.2.4.5 Add Bookmark

3.2.3.2.4.5.1 Alternative Flow 1

5. User enters incorrect or incomplete coordinate values and tabs on “complete adding” button.

6. A message window is showed to the user, informing him/her that the coordinates values are incorrect.

7. The user is redirected to the main “Add Destination Points Screen”.

3.2.5 Show Path

3.2.5.1 Background Information

The software's main goal is to generate shortest path between seleted points and to show result to the user on the map. The process and relates functionalties about adding destination points is explained in the previous section. The software will present several options to the user about showing shortest path. The user will be able to choose to see whether the shortest path or path with minimum expected travel time or the path with

minimum expected fuel consumption. He/she will also be able to reset the path. If that functon is used as a result the generated path will dissappear on the map.

3.2.5.2 Stimulus/Response Sequences

3.2.5.2.1 Diagram

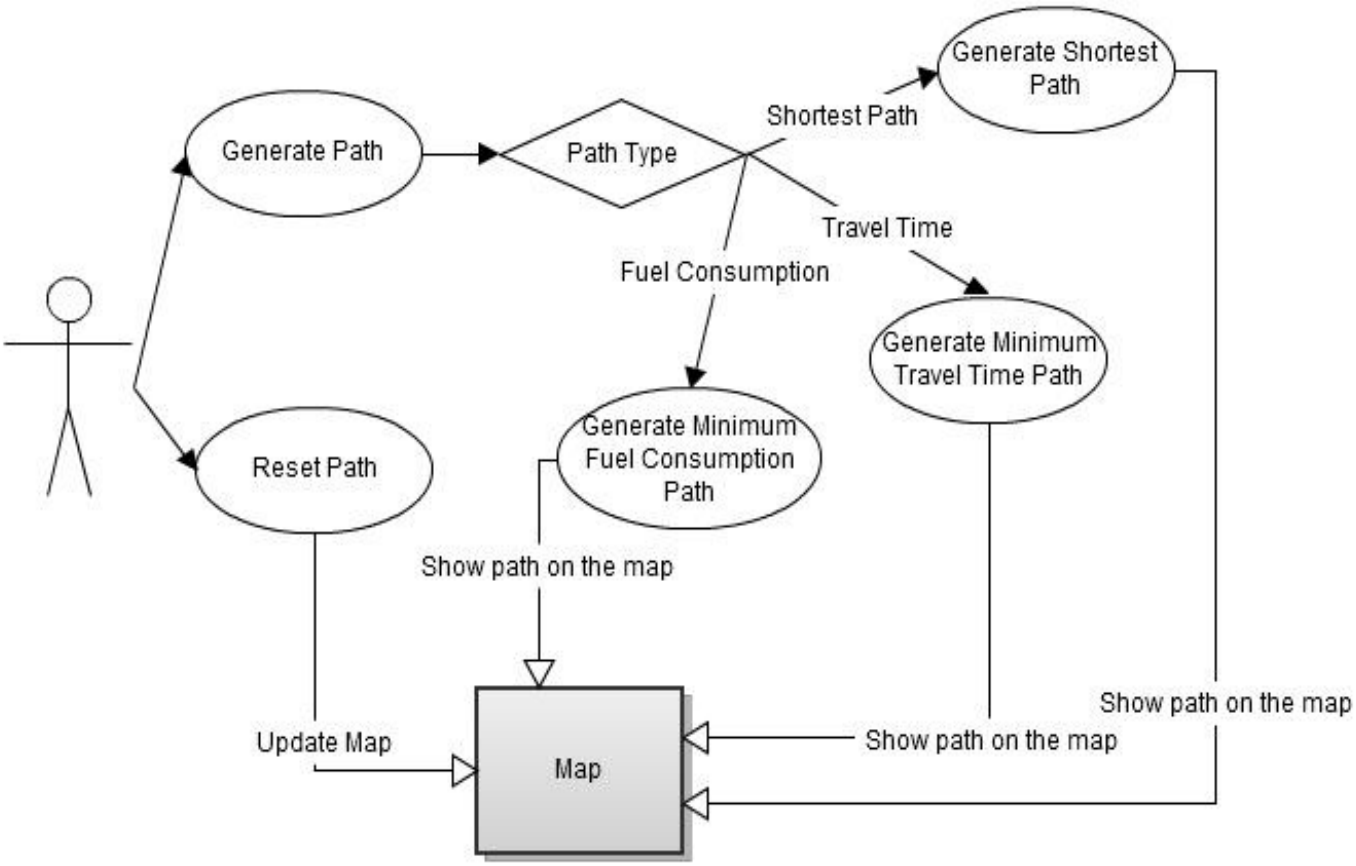


Figure 6: Use Case Diagram for Show Map

3.2.5.2.2 Description

Aim	To calculate and to show the shortest path on the map
Precondition	User have entered points to the list
Trigger	User tabbing on "generate path" button

3.2.5.2.3 Normal Flow of Events

3.2.5.2.3.1 Generate Path

1. User tabs on generate path button

2. A new menu pops with three different options: shortest path,

minimum travel time path, minimum fuel consumption path.

3. User tabs on one of the options.

4. The software make neccessary calculations and displays the path on the map.

3.2.5.2.3.2 Reset

1. After path is generated a reset button appears on the screen.

2. User tabs on the button.

3. The generated path on the map dissapears.

4. Generate path button reappears.

3.2.5.2.4 Alternative Flow of Events

3.2.5.2.4.1 Generate Path

3.2.5.2.4.1 Alternative Flow 1

4. If one or more of the destination points is not in the range of area where the selected vehicle can travel, the generate path algorithm will ignore that point. The resulting path will be shown on the map.

5. An information box will be shown on the map, informing the user about the situation.

6. The user tabs "OK" button on info box.

7. The user is redirected to "Add Destination Points Screen".

3.3 NON-FUNCTIONAL REQUIREMENTS

In this section we will present performance requirements and design constraints of Patika Project.

3.3.1 PERFORMANCE REQUIREMENTS

Every feature of the system described in section 3.2 will be able to run on 1 GHz Application Processor. Targets may be moving or non moving, multiple or singular, these facts should not change the performance of the application distinctly. There will be no server performance requirements because this application will be developed for

single user purpose only. There will be no multiple user handling and no need for any network.

3.3.2 DESIGN CONSTRAINTS

The Patika Application will follow the following design constraints:

- Programming language that will be used during the project is Java, since we will be developping this application for Android platforms and Java is the native language of Android.
- The system shall run on a mobile device so the processor speed, the quality of the graphics card should be enough to be able to run the application.
- Since the system will run on a single machine and it will not be dependent on a bigger system or any other computers, there will not be any security or reliability problems.
- Since we will be using dynamically changing maps, our application should not wait too much to produce the new map, and if there an error occurs, user shall be notified immediately.

3.2.3 SIMPLICITY

Regarding user interface, an important design goal is keeping it simple. User interface should not be complicated. Besides, the user should not need to do anything for basic usage of the system, that is showing targets and paths.

3.2.4 ACCURACY

Accuracy of the system is important for the user. For this reason, the system should have 100% accuracy, meaning that it should determine all paths correctly.

4. Data Model and Description

This section will give information about the general data objects, their attributes and the complete data model.

4.1 Data Description

This section will give information of application's data objects and relationships between these objects.This section also contains attributes and a dictionary for these attributes for simplicity.Moreover, there will be the complete data model of these data objects.

4.1.1 Data Objects

The main data objects of application are Map, Point and Vehicle.First object which is Map will be used object of visualised map which is main part of application.Map object has MapType, MapFile, MapWidth and MapHeight attributes.Second object is Point will be used visualise coordinates and places of user or targets.Point has Longtitude,Latitude, Altitude, BookmarkFlag, BookmarkName and RecentFlag attributes.Last object is Vehicle which represent user transportation vehicle for movement.Vehicle has VehicleName, VehicleMaxSpeed, VehicleMaxRamp, VehicleMaxFuelAmount and VehicleMovementAreas attributes.

4.1.2 Relationships

As mentioned in previous section, application will have three main objects which are Map, Point and Vehicle.From the relationship perspective, Map contains both Point and Vehicle.Parent objects of Point and Vehicle is Map so every Map contains multiple Points that is one of them for represent location of user and the others for target points.Vehicle on Map represents vehicle of user so that Vehicle has a Point object which represent location of Vehicle and user.

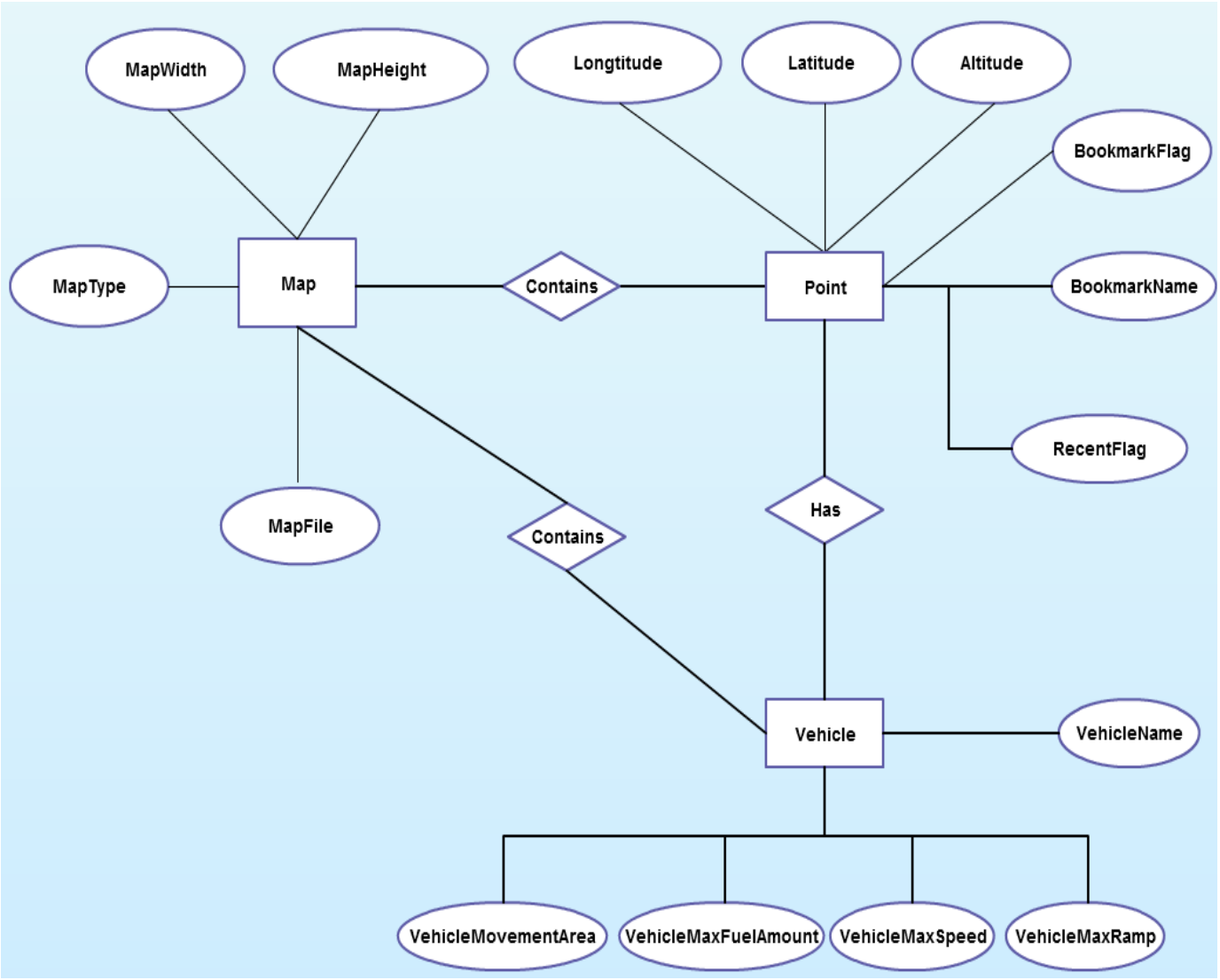


Figure 6: ER Diagram

4.1.3 Complete Data Model

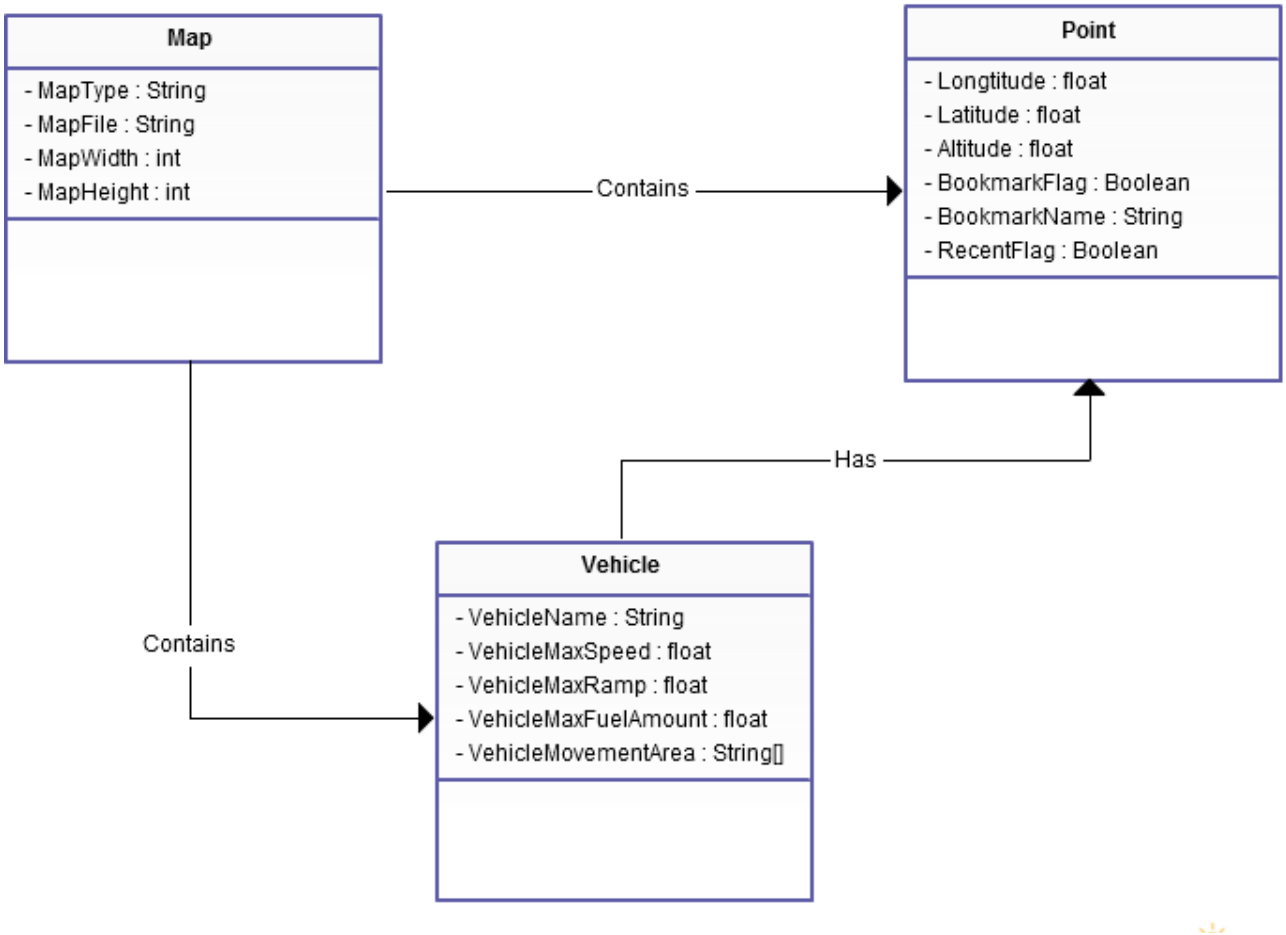


Figure 6: Class Diagram

4.1.4 Data Dictionary

4.1.4.1 Map

a) MapType : Defines type of map which is Terrain.

b) MapFile : DTED2 file of map which is used to get data to visualise map.

c) MapWidth : Width length attributes of map.

d) MapHeight : Height length attributes of map.

4.1.4.2 Point

a) Longtitude : Defines longtitude of point

b) Latitude : Defines latitude of point

c) Altitude : Defines altitude of point.

d) BookmarkFlag : Specify whether point is bookmarked or not.

e) BookmarkName : Determines name of bookmark point if the point is bookmarked

f) RecentFlag : Specify whether point is recently added or used.

4.1.4.3 Vehicle

a) VehicleName : Defines a unique name of vehicle.

b) VehicleMaxSpeed : Defines maximum speed of vehicle.

c) VehicleMaxRamp : Defines maximum slope of area that vehicle can continue to move.

d) VehicleMaxFuelAmount : Defines maximum amount of fuel that vehicle can contain.

e) VehicleMovementArea : Specify types of area that vehicle can move.

5. Behavioral Model and Description

We will first give a description about behaviour of the software and then we will ilustrate the behaviour with the state transition diagram. The software has nine major states and they can be listed as: “Obtainment location information”,” Waiting for Wifi/GPS”, “Update Location”, “ Main Screen”, “Vehicle Screen”, “Update Vehicles”, “Add Destination Points Screen”, “Update Shorthest Path” and “Map Operations”. Morover there are start and end stages.

5.1 Description of Software Behaviour

The first state of the software is start state which directy invokes “Obtainment location information” state. In that state, the main goal in order to continue to normal procedure of the software to get the location information. However, in order to get location information the software needs data that will come from Wifi or GPS. Therefore, in that state “ask user to open Wifi/GPS” event will invoked and it will lead the software to “Waiting for Wifi/GPS” state. In that state, the application will wait user to open Wifi/GPS, after user opens one or another, “Wifi/GPS opened” event is invoked. Then, current state becames again “Obtainment Location Information” and, because Wifi or GPS is opened, “Location Determined” event is invoked and as a result the next state becames “Update Location”. This state leads to “Main Screen” with updating location of the user in the

height map taht will be displayed on the main screen. The software will generally display map and graphical user interface elementsi meanwhile will be waiting for user actions. According to the user actions, there is several states that user actions can lead. If input coming from user leads the software to the “Vehicle Screen”, that state will be waiting for user to perform actions like “add/delete/select vehicle”. That actions will lead to “Update Vehicles” state. The system will update vehicles in database and present results to user by invoking update vehicles event and the system will return to “Vehicle Screen” state. If the user perform “Return to main screen” action, the software will return to the “Main Screen” state. The zoom in/out, move to left/right/top/bottom actions are very important for system, because it will display maps and shorthest paths between selected points dynamically. If user performs such actions, the system will enter to map operations state. This state will lead the “Main Screen” state after updating map according to the performed actions by user. Most important loop in the state machine is one performing shorthest path calculations. “Add Destination Points Screen ” state became current state when performs “enter to add destination points screen” action. After user finishes to add points and demand generation of shorthest path, the system enters to “Update Shorthest Path” state. After generation is completed, “Update path on the map” actions is invoked and the system returns to the “Main Screen” state. In order to end the software user must perform the “Quit ” action. This action can be performed when system is in “Main Screen State” or “Obtainment Location Information” states.

5.2 State Transition Diagrams

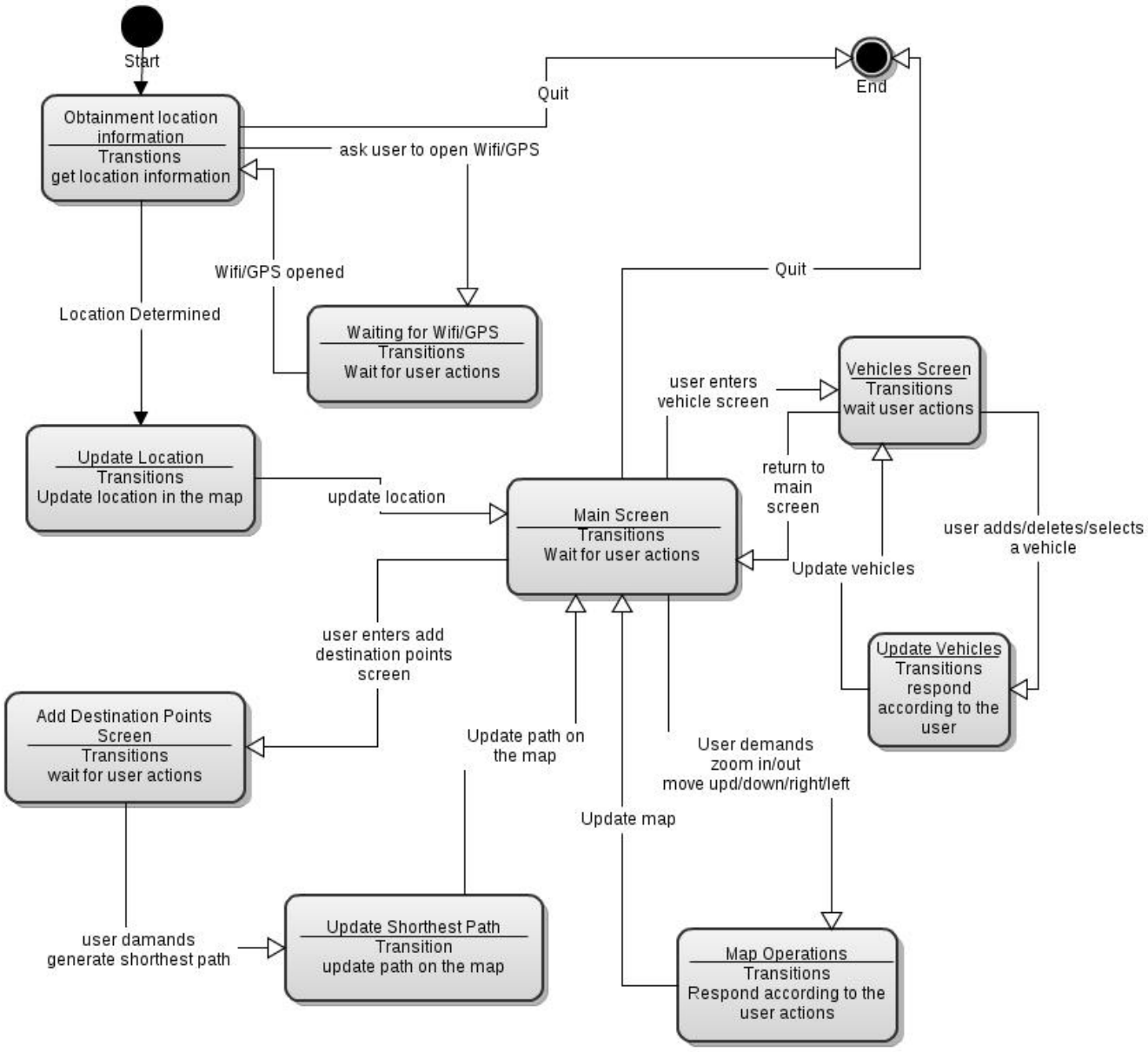


Figure 5 – State Transition Diagram

6. PLANNING

In this part of the document, the structure of the team responsible from the project, the basic schedule, and the process model will be presented.

6.1 Team Structure

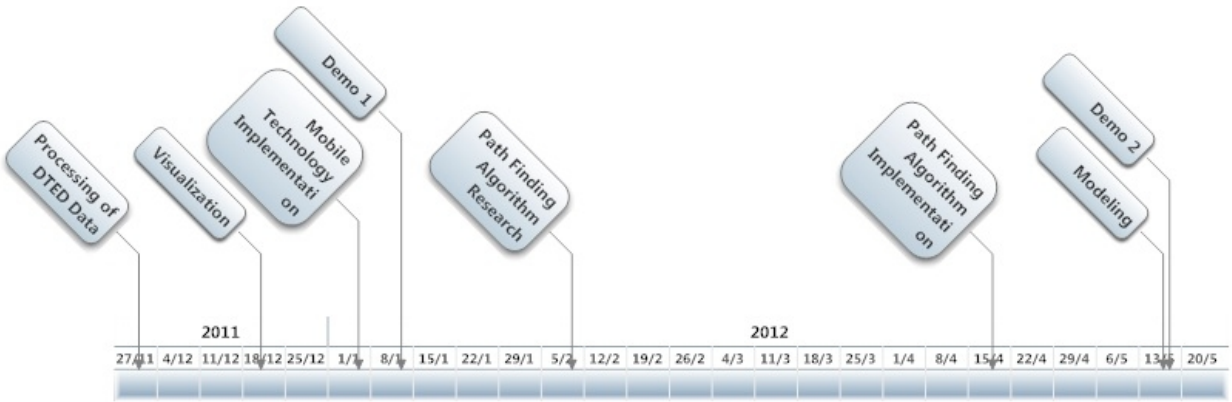
Pınar Yılmazer: Researcher, Software Engineer, Path Algorithm Developer

Bahar Şevket: Researcher, Software Engineer, Graphics Developer

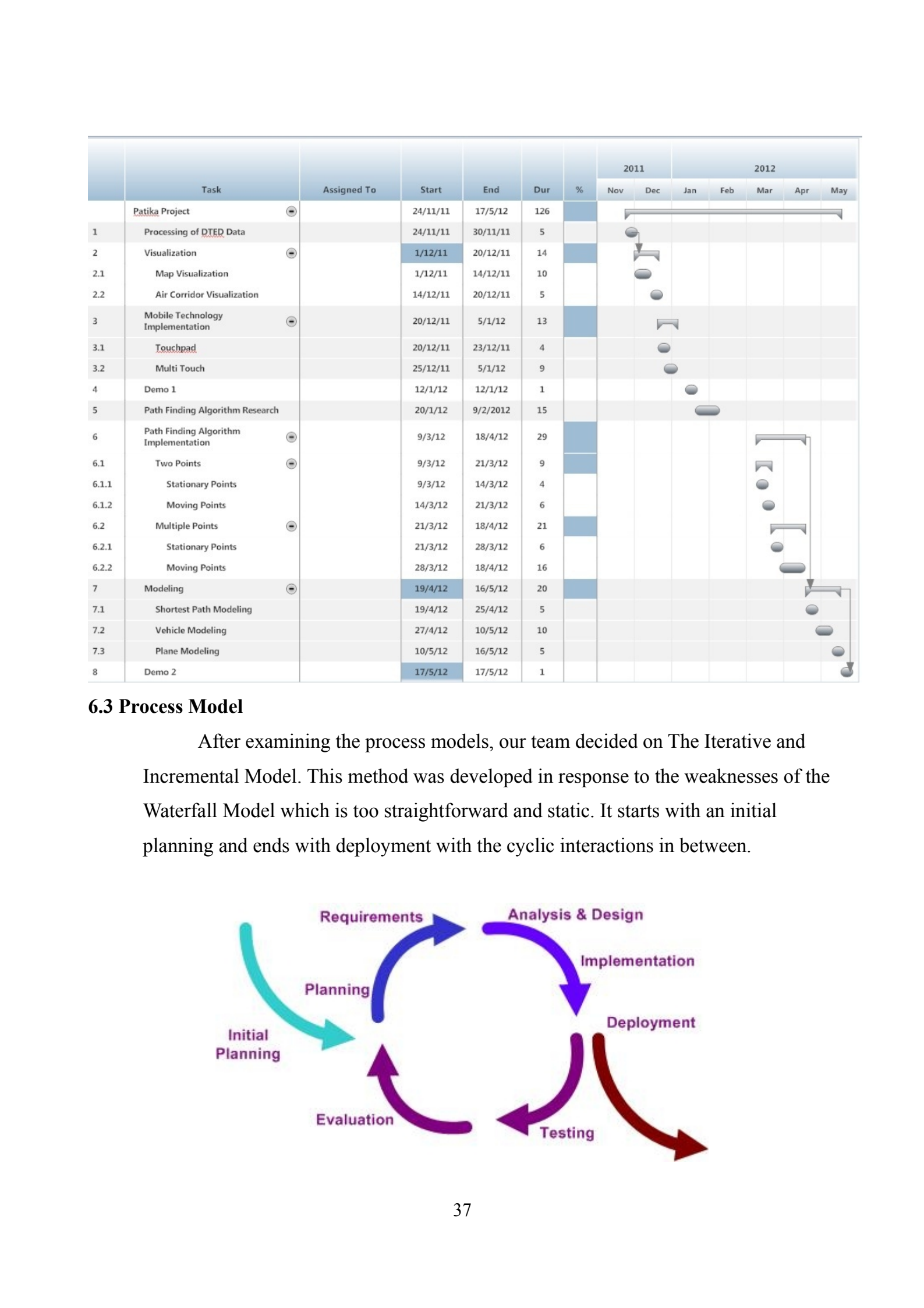
Gözde Özcel : Researcher, Software Engineer, Path Algorithm Developer

Levent Oral : Researcher, Software Engineer, Mobile Platform Developer and Integrator

Since the scope of this project is extensive, we divided it to three subparts to better specialize on individual parts. We plan to do this project on the principle of equal job / hardship for everybody. We don't have a leader, but Levent is our contact between the group and the company.



	Task	Assigned To	Start	End	Dur	%
	Patika Project		24/11/11	17/5/12	126	
1	Processing of DTED Data		24/11/11	30/11/11	5	
2	Visualization		1/12/11	20/12/11	14	
2.1	Map Visualization		1/12/11	14/12/11	10	
2.2	Air Corridor Visualization		14/12/11	20/12/11	5	
3	Mobile Technology Implementation		20/12/11	5/1/12	13	
3.1	Touchpad		20/12/11	23/12/11	4	
3.2	Multi Touch		25/12/11	5/1/12	9	
4	Demo 1		12/1/12	12/1/12	1	
5	Path Finding Algorithm Research		20/1/12	9/2/2012	15	
6	Path Finding Algorithm Implementation		9/3/12	18/4/12	29	
6.1	Two Points		9/3/12	21/3/12	9	
6.1.1	Stationary Points		9/3/12	14/3/12	4	
6.1.2	Moving Points		14/3/12	21/3/12	6	
6.2	Multiple Points		21/3/12	18/4/12	21	
6.2.1	Stationary Points		21/3/12	28/3/12	6	
6.2.2	Moving Points		28/3/12	18/4/12	16	
7	Modeling		19/4/12	16/5/12	20	
7.1	Shortest Path Modeling		19/4/12	25/4/12	5	
7.2	Vehicle Modeling		27/4/12	10/5/12	10	
7.3	Plane Modeling		10/5/12	16/5/12	5	
8	Demo 2		17/5/12	17/5/12	1	



6.3 Process Model

After examining the process models, our team decided on The Iterative and Incremental Model. This method was developed in response to the weaknesses of the Waterfall Model which is too straightforward and static. It starts with an initial planning and ends with deployment with the cyclic interactions in between.

7. CONCLUSION

This software requirements specification document gives information about the project Patika. The scope of this document includes a brief introduction to the technologies used in similar applications and market research, analysis of product perspective, functionalities, constraints and dependenices. There are also detailed explanations on specific requirements of the project, and the functional requirements part to better explain the basic working principle of the Patika. There is the part of non-functional requirements, where we aimed to clearly state the requirements and constraints of mobile platform software development. Finally, we gave an estimation on the timeline of the project, presented the team organization and process model of the project.