myTaxiService

Requirement **A**nalysis and **S**pecification **D**ocument

Politecnico di Milano

Cattaneo Michela Gaia, matr. 791685
Barlocco Mattia, matr. 792735

November 6, 2015

Contents

1 Introduction

1.1 Purpose

This document represent the Requirement Analysis and Specification Document (RASD). This document aims at describing the system in terms of functional and non-functional requirements, modeling the system according to the real needs of the customer and showing the constraints and the limit of the software. This document serves as a contractual basis between the customer and the developer.

1.2 Actual system

The government of a large city has commissioned a system that is able to manage and optimize the taxi service.

1.3 Scope

The system aims at simplifying the passengers access and guaranteeing a fair management of taxi queues. In order to manage the queue optimization, the city is divided in zones, each one associated to a taxi queue, and the system computes the taxi distribution basing on their GPSs information.
The system allows the user of the application to request a taxi in an area of the city, even without logging in. As soon as a taxi driver takes the request, the application notifies the user of the ID of the incoming taxi and the waiting time. In addition, the user can register, by filling in a form, and make reservations: he only needs to indicate the origin and destination of the ride and the exact time (at least 2 hours in advance) and the system reports that a taxi will be there to pick him up.
On the other hand, the taxi driver always has to notify his availability and let the system know if he is going to take a certain request by a passenger. If the driver is not able to confirm, the system will forward the request to the second in queue and move this taxi driver in the last position of the queue.

1.4 Actors

The actors participating in the system are:

- Visitor: a person who can see the home page of the application, he can register, look up for information about the system and request a taxi, after giving name and phone number.
- Passenger: a regular registered user who wants to use the application in order to request a taxi in a specific zone or make a reservation for a given time.
- Taxi Driver: a registered user who logged in with a specific ID. This area is reserved to taxi drivers who needs to use the application in order to inform the system of their availability or about what request they are going to take care of.

1.5 Goals

A user should be able to:

- Request a taxi.
- Payment method.
- Check the waiting time.
- Visualize the information about the system.
- Sign up into the system.

A passenger should be able to:

- Reserve a taxi by specifying the origin and the destination of the ride, at least two hour before the ride.

The taxi driver should be logged in with a specific ID and should be able to:

- Change his state from available to unavailable and vice versa.
- Visualize the queue of his area.
- Answer positively or negatively to the request that the system forwards to him.
- Visualize the position of the user he needs to reach.

The system should be able to:

- Assign each taxi to the queue of the designated area, based on GPS position.
- Forward the request to the first taxi in the queue.
- If the first taxi of the queue doesn't take the request, the system will send the request to the second taxi of the queue and, at the same time the first taxi will be moved in the last position of the queue.
- If the passenger wants to reserve a taxi, the system will forward the request to the first available taxi in the area ten minutes before the ride.

1.6 Domain properties

We suppose that the following properties hold in the analyzed domain.

- We assume that there are not previous implementations managing this service.
- If a visitor requests a taxi, he will already be where the taxi driver expects him to be.
- If a passenger reserves a taxi, he will be where and when the taxi driver expects him to be.

- If a passenger reserves a taxi, he will not cancel the reservation less than 2 hours earlier.
- If a taxi driver inform the system that he is available, he actually is.
- If a taxi driver notifies that he is going to take a certain request, he will always do.
- When a taxi driver finishes a ride, he instantly changes his state from unavailable to available.
- If a user requests a service from the system, it will be within the borders of the city.

2 Overall Description

2.1 Product perspective

The product consists in a web application and a mobile app, based on the web. There is a server part, that manages all the requests of the users and the answers and the availability of the taxi drivers, computing their zone distribution. There are several clients (visitors, passengers or taxi drivers) interacting with the server by using a graphical user interface, but there is no implementation of interfaces for administration, in fact the application is only user-based.

2.2 User characteristics

The user can be a person who wants to request a taxi or access the application in order to use additional functionalities, such as make a reservation or easily request a taxi without having to give his information every time he exploits the service. The application is used also by the taxi drivers who needs to receive the requests of the customers and to interact with the system, responding to the calls and updating their availability and their GPS position . The user must have access to the Internet.

2.3 Constraints

2.3.1 Regulatory policies

The system must ensure the privacy of the users, without publishing their personal information.

2.3.2 Hardware limitations

The system doesn't work if the smartphone is not powerful enough or it has not the GPS detector.

2.3.3 Interfaces to other applications

The system interfaces with two applications:

- Google maps to determine the area of a taxi or the position of a user.
- Facebook if a user wants to sign in with his Facebook account.

2.3.4 Parallel operations

The system supports parallel operations of different clients accessing the application and making requests or reservations of taxis.

2.3.5 Documents related

- Requirements and Analysis Specification Document (RASD).
- Design Document (DD).
- User's Manual.
- Testing report.

2.4 Assumptions

- User can only make one request or reservation at a time.
- The system changes the state of a taxi driver from available to unavailable if he responds positively to a request, but not vice versa.
- If the passenger makes a reservation less then two hours in advance the system will not accept it.
- The system is able to manage the taxi queues in a way that there is always a taxi driver available for the reservation within ten minutes before the ride.
- The taxi driver has two minutes to respond to the request.
- The profile of the passengers are private, no other users can visualize it.
- The taxi driver receives the phone number and the position of the user if he take the request.

2.5 Future possible implementations

The application must be open to future implementations. Future possible implementations are:

- Taxi sharing: this is an additional functionality that allows different passengers to share the taxi with other passengers. The system must be notified that the passenger wants to use the taxi sharing option and it will ask him to indicate the destination of this rides. When it is all settled, the system is able to check if there are other persons in the same area that wants to reach the same destination, informing the taxi driver and the passengers. This can be a convenient solution, as the cost of the ride would be equally shared among the passengers.
- Price transparency: this is an additional functionality that allows the passenger to calculate the price of the ride in advance, inserting its origin and destination.
- Meet the driver: this is an additional functionality that lets the passenger know which taxi driver has answered his request. The system will send the passenger the profile of the designated driver with his name and data, so that the he would be able to recognize him or even call or text him if necessary.
- Give us feedback: this is an additional functionality that allows the passenger to rate the driver after the ride and leave comments about it.
- Easy to pay: this is an additional functionality that allows the passenger to automatically pay with the credit card associated with his account. The system will send him an e-mail with the receipt of payment.

3 Specific Requirements

3.1 External interfaces requirements

3.1.1 User interfaces

3.1.2 API interfaces

3.1.3 Hardware interfaces

This project does not support any hardware interfaces.

3.1.4 Software interfaces

3.1.5 Communication interfaces

3.2 Functional requirements

1. **Registration of the visitor.** The visitor exploits the sign up functionality, he fills in the registration form, by inserting his personal information. Some fields are mandatory, such as: name, surname, password, e-mail address and telephone number.

2. **E-mail confirmation.** It is necessary to confirm the e-mail address, following the link provided in the e-mail received after the registration and before the first log in.

3. **Look up information about the system.** The user can see the data of the system, in particular the description of the application, the contacts of the managers of the system and information about the investors.

4. **Log in of the passenger.** The passenger inserts username and password in an input form.

5. **Log in of the taxi driver.** The taxi driver inserts e-mail, ID and password in an input form.

6. **Retrieve password.** The system allows the user or the taxi driver to recover his password by receiving a recover e-mail.

7. **Request a taxi.** The user can make taxi requests enabling the GPS position so that the system can determine his position and can send the call to the first taxi driver of the queue of the designated area. The visitor can exploit this service in the home page of the application, inserting his name and telephone number. The passenger can use this service without inserting any further information.

8. **Reserve a taxi.** The passenger can reserve a taxi by indicating the origin and the destination and the time of the ride.

9. **Payment method.** The user can choose whether to use cash or credit card as payment method. The visitor needs to provide his credit card data every time he chooses this second method, while the passenger only needs to add his credit card data only if this is the first time he chooses this payment method. The system deduct the money from the credit card

only when the ride is over, sending also the receipt of the payment by e-mail.

10. **View or modify the profile.** The passenger can see or modify his profile. He can edit or add his profile picture, other personal information and the credit card data. He can also update his personal information.

11. **Notification panel.(user)** Perch un pannello notifiche? basterebbe che l'app rimanesse in attesa di un taxi con un loading screen e che uscisse la risposta nel caso qualcuno prenda la chiamata. COME SCRIVERE QUESTA COSA?

12. **Waiting time panel.** The user can see the waiting time panel from the time he makes the request of a taxi until the taxi arrives. This consists in a display showing the time the user has to wait till the taxi arrival.

13. **Availability button.** The taxi driver can switch his state from available to unavailable and vice versa. He sets his state available when he has just finished a ride or in general when he is able to take a request from a user. He doesn't need to set his state unavailable when he takes a call, as it is already managed by the system, but this setting is necessary when he is not able to take requests or when he finishes work.

14. **Notification and position panel.(taxi)** The taxi driver can see the requests by a user as notifications. The system sends him a notification with the information on the user and his position, the taxi driver can accept the request or refuse it.

15. **Position in the queue.** The taxi driver can see his position in the queue in his area, so that he is always able to know if there can be any incoming requests. The system constantly updates (every minute) the position of all drivers of all areas.

3.3 Non-functional requirements

3.3.1 Performance requirements

The system is composed of a server side hosting the database, where all the data of the passenger and of the taxi driver are stored, and it handles all HTTP request. There is not a great amount of data to manage, there are only profile information of the passengers, of the taxi drivers and the log of the activities, then the size of the database is not a constraint for the system.
To supply a suitable service, the system has to be reactive and able to answer to a high number of simultaneous request.
There are no time constraints that depends on the system, the time to process a transaction is less then one second. The only time constraint would be the reply message of the taxi driver that depends on his reflexes.

3.3.2 Software system attributes

1. Reliability: the server of the system is on 24 hours on 24, every day of the week to guarantee a suitable service. The system allows administrators to fix some problems without compromise the functionality of the system.

2. Availability: to guarantee the availability of the data, there is a backup system on a secondary database and one on an external storage.

3. Security: the privacy of the exchanged data between server and client is guaranteed by SSL encryption. The passwords and the sensitive data which are stored on the database are protected by MD5 encryption.

4. Portability: the client side of the system is compatible with the major platform on the market (e.g. Android, iOS) and any device that support a browser and a GPS system.

4 Scenarios Identifying

4.1 Scenario 1 [PASSWORD RECOVERY]

USER1 has registered in the application when he visited the city a few months ago, now he is back and needs to make a taxi reservation for the afternoon. But USER1 is really absent-minded and is not able to remember his password. He is very happy to see the button "password recovery" and he press it: this leads to another page where the system tells him that an e-mail for the recovery has been sent to his e-mail address. He checks his inbox and follows the link to set his new password. Now he can easily log in and make his reservation.

4.2 Scenario 2 [ADD INFORMATION TO PROFILE]

USER2 often uses myTaxiService for the taxi requests, as it is a fast and reliable service. He is a very lazy person and does not want to register, as the application allows visitors to easily request a taxi, but he is very tired of inserting his credit card data every time he is out of cash, so he finally make up his mind. He registers and he adds his credit card data to his profile along with a funny profile picture of a kitten. He is satisfied and he can request or reserve a taxi without even worry about the money spent.

4.3 Scenario 3 []

USER3

4.4 Scenario 4

4.5 Scenario 5

5 UML Models

5.1 Use Case

5.2 Class diagram

5.3 Sequence diagram

5.4 Statechart diagram

6 ALLOY Modeling

6.1 Data type

6.2 Abstract entity implementation and signature

6.3 Fact

6.4 Assert

6.5 Predicates

6.6 Result

6.7 Generated world

7 Appendix

7.1 Glossary

In order to avoid ambiguity, some words, that will be often used in this document, will be given a precise definition of what will be the meaning in the contest of this project.

- User: a person who requests a service from the system. It can be a visitor or a passenger.
- Visitor: a person who is not registered in the application.
- Passenger: a person who is registered in the application.
- Taxi driver: a taxi driver who access the application with a specific ID.
- Request: the request of a taxi in a certain zone and place of the city made by a user.
- Reservation: the reservation of a taxi in a certain zone, place and time that can be made only by passengers.

7.2 Software and tool used

- TeXstudio (http://www.texstudio.org/): to redact and to format this document.
- draw.io (https://www.draw.io/):to create Use Case Diagrams, Sequence Diagrams, Class Diagrams and Statechart Diagrams.
- Balsamiq Mockups(http://balsamiq.com/products/mockups/): to create mockups.
- Alloy Analyzer(http://alloy.mit.edu/alloy/): to create ALLOY models.

7.3 Hours of work

Time spent redacting this document:

- Cattaneo Michela Gaia: ~13 hours of work.
- Barlocco Mattia: ~13 hours of work.