Politecnico di Milano
A.Y. 2015/2016

My Taxi Service

Requirement Analysis and Specification Document

Bernardis Cesare matr. 852509 Dagrada Mattia matr.852975

November 6, 2015

Contents

1 Introduction

1.1 Purpose

The main goal of this document is to completely describe the system in terms of functional and non-functional requirements, to analyse the real need of the customer modelling the system, to show the constraints and the software limits and simulate the typical use cases that will occur after the development. This document is intended to all developers and programmers who have to implement the requirements, to the system analysts who want to integrate other system with this one, and could also be used as a contractual basis between the customer and the developer.

1.2 Actual system

The government of a large city wants to optimize its taxi service. We suppose that the actual taxi service is based on simple phone calls from customers to the taxi's call centre.

1.3 Scope

The aim of this project is to create a brand new taxi application that is used by both the taxi drivers and the passengers to access the taxi service. Passengers can access the service either via mobile or web application. They can request a taxi without having to register to service but they have to insert personal information while registered passengers, once logged in, can directly request a taxi. The system confirms a taxi request by sending the passenger the taxi code only when a taxi is found and an estimated arrival time. Registered passengers can also make taxi reservations and they will receive the code as soon as a taxi is found available. Taxi drivers can access the service only from mobile application. Once logged in they can give the system their availability, or revoke it, and they will receive from the system ride requests that they can either accept or refuse. The login interface of the application is shared by passengers and taxi drivers. The system has the city map divided into areas of $2km^2$ and holds a taxi queue in each area. It receives GPS coordinates from taxis, it assigns them to their corresponding area queue, placing them in the last position. Following a FIFO (First In First Out) logic, the first taxi receives a request and is then removed from the queue. In case of rejection, the taxi is placed in the last position of the queue.

1.4 Actors

Guest a guest is able to request a taxi without having to be registered to the service, but simply adding some essential personal information. A

guest can register to the service filling in a registration form, becoming a Registered Passenger.

Registered Passenger a registered passenger, once logged in, is able to request a taxi without having to add additional personal information. A registered passenger can also make reservations, specifying origin and destination of the ride, at least 2 hours before the desired time for the ride.

Taxi Drivers a taxi driver, once logged in, has a different user interface than registered user, from which he will receive ride requests that he can accept or refuse. He will also be able to give his availability, which means that he is willing to pick up ride requests.

1.5 Goals

These are the goals of MyTaxiService application:

- Permit a guest to register to the service.
- Permit a guest to request a taxi.
- Permit a guest to sign in and become a registered passenger
- Permit a registered passenger to require a taxi.
- Permit a registered passenger to make a reservation.
- Permit a registered passenger to cancel a reservation.
- Permit a taxi driver to give the system his availability.
- Permit a taxi driver to revoke his availability.
- Permit a taxi driver to accept or refuse a ride request.

1.6 Reference Documents

- Specification Document: MyTaxiService Project A.Y.2015-2016.
- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.

1.7 Document overview

This document is structured as following:

1. **Introduction**: this section represents a generic description of the project, highlighting actors and goals.

2. **Overall Description**: this section gives further informations about the project, focusing more on what are the assumptions made about the software and its constraints.

3. **Requirements**: this section lists the requirements of the software, both functional and non-functional ones.

4. **Scenarios**: this section shows some typical scenarios.

5. **UML Models**: this section contains some typical use cases and class diagrams.

6. **Alloy Modelling**: this section contains Alloy code and Alloy worlds.

7. **Appendix**: this section contains extra information and also which software and tools has been used to write this document.

2 Overall Description

2.1 Product perspective

We want to release both a web and a mobile application, avoiding deep integrations with other existing systems in order to keep an high level of compatibility, especially the mobile application, that we want to release for all major mobile operative systems. The applications will not have any internal interface for administration but they will be only user based. The application will provide APIs for a faster, safer and better integrated implementation of new features, based on the basic functionalities offered by the system.

2.2 User characteristics

We expect to have two different types of user with distinct experiences.

Passenger who needs to call a taxi for a ride (that is occasional, threated as a guest, or habitual, who can register himself to speed up request submissions). This user must have access to the Internet and be able to use a web browser or have installed our mobile application on his smartphone;

Taxi Driver who wants to offer its service through our system. This user must have access to Internet and have installed our mobile application on his smartphone.

2.3 Constraints

2.3.1 Regulatory policies

Personal information (locations included) obtained from the user will be stored in our databases if strictly necessary, to provide the best possible experience with our service. No information will be disclosed to any other company or used for any other purpose.

2.3.2 Hardware limitations

MyTaxiService's web application will be available on every device with an Internet connection and a browser installed. The mobile application (necessary for taxi drivers) will have the following requirements:

- 256MB total RAM
- 50MB of free space on Disk
- Internet Connection

- Operative System:
  - Android 2.3 "Gingerbread" or later
  - Windows Mobile 7.0 or later
  - iOS 6.0 or later

It is also recommended to activate the GPS for a optimal localization.

2.3.3 Interfaces to other applications

MyTaxiService will provide an API library to allow external developers to integrate their applications with our system.

2.3.4 Parallel operation

The system must support parallel operations from different users. Information integrity is fundamental to provide a first-rate service, so the mechanism of supply and demand will have to be highly affordable.

2.3.5 Documents related

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

2.4 Assumptions and Dependencies

- There is not an administrator or a privileged user. We think that is not necessary a hierarchy of users to keep the system safe;
- There is not any dependence between users;
- Guest user is identified through his public IP address;
- The position of the user can always be determined, via browser or GPS (see HTML5 Geolocation), and has a good approximation. Otherwise the service will not be accessible;
- A taxi driver can access the service only if he is signed up. During registration he will have to provide, in addition to the usual personal information, the identification number of his taxi;
- Through the ID number of a taxi it is possible to retrieve some informations (like the location) of the taxi itself;

- A taxi driver can give and remove availability in every moment;
- If a taxi driver does not accept a call within 1 minute after the notification, a rejection will be automatically recorded in the database and the request will be forwarded to another taxi;
- After 3 consecutive rejections (voluntary or involuntary) the taxi availability will be revoked automatically;
- When a user (registered or guest) requests a taxi, his position is taken automatically. If he does not agree, his request will not be forwarded;
- After a request, a user (registered or guest) can not make other requests in the next 30 minutes;
- A user must be signed up to make a reservation;
- Reservations must be taken at least 2 hours in advance from the time of taxi's request, otherwise the reservation will not be accepted;
- A user that registered a reservation for a taxi can not make requests within 30 minutes before the reservation;
- A reservation can be cancelled until 10 minutes before the request time.

2.5 Future possible implementation

The taxi sharing option: this means that the user is ready to share a taxi with others if possible, thus sharing the cost of the ride. In this case the user is required to specify the destination of all rides which he/she wants to share with others. If others are willing to start a shared ride from the same zone going in the same direction, then the system arranges the route for the taxi driver, defines the fee for all persons sharing the taxi and informs the passengers and the taxi driver.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 API Interfaces

For taxi waiting time and other minor features we decided to use Google Maps API. It is a very powerful library that perfectly fits our needs. When a taxi accepts a request, the system retrieves its position and, through those APIs, is able to process either the hypothetical route that the taxi will follow to reach the passenger's position and, consequently, the arrival time. The system will, then, forward those information to the passenger.

3.1.2 Hardware Interfaces

Our project includes a web and a mobile application, so it does not require any external hardware interface.

3.1.3 Software Interfaces

- Database Management System (DBMS):
  - Name: MySQL.
  - Version: 5.6.21
  - Source: http://www.mysql.it/
- Java Virtual Machine (JVM).
  - Name: JEE
  - Version: 7
  - Source: http://www.oracle.com/technetwork/java/javaee/tech/index.html
- Application server:
  - Name: Glassfish.
  - Version: 4.1.
  - Source: https://glassfish.java.net/
- Operating System (OS).
  - Application must be able to run on any SO which supports JVM and DBMS specified before.

3.1.4 Communication Interfaces

Protocol	Application	Port
TCP	HTTPS	443
TCP	HTTP	80
TCP	DBMS	3306 (default)

3.2 Functional Requirements

3.2.1 Permit a guest to register to the service

R1: Guest can access registration form.

R2: Guest must not be already registered to perform registration process.

R3: Guest must choose a unique username (never used by other users)

R4: Guest can not sign up twice but only once for session.

R5: During registration must provide all necessary information

3.2.2 Permit a guest to request a taxi

R1: Guest must provide basic personal information

R2: Guest must agree to provide his position

3.2.3 Permit a guest to sign in

R1: User must be already registered to sign in.

R2: User must provide login information to complete the operation.

R3: The couple username and password inserted during login process must be correct (the same couple inserted during registration).

R4: Wrong credentials will not grant the access.

R5: After the login operation, the user will be treated as a registered passenger.

R6: After the login operation, the registration form will not be available for the current session.

3.2.4 Permit a registered passenger to require a taxi

R1: User must be already registered and logged in the application.

R2: User must agree to provide his position.

R3: No additional information is required.

3.2.5 Permit a registered passenger to make a reservation

R1: User must be already registered and logged in the application.

R2: User must make a reservation at least 2 hours before the request time.

R3: User must provide time, origin and destination of the ride at the moment of the reservation.

R4: User must confirm the reservation process.

R5: The reservation can be deleted by the user himself.

3.2.6 Permit a registered passenger to cancel a reservation

R1: User must be already registered and logged in the application.

R2: User can delete only a reservation made by himself.

R3: User can not delete a reservation within 10 minutes before the time of the request.

R4: User must confirm the deleting process.

R5: Deleting process is not reversible.

3.2.7 Permit a taxi driver to give the system his availability

R1: User must use the mobile application.

R2: User must be already registered and logged in the application.

R3: User must be a taxi driver.

R4: The taxi driver must be unavailable.

3.2.8 Permit a taxi driver to revoke his availability

R1: User must use the mobile application.

R2: User must be already registered and logged in the application.

R3: User must be a taxi driver.

R4: The taxi driver must be available.

3.2.9 Permit a taxi driver to accept or refuse a ride request

R1: User must use the mobile application.

R2: User must be already registered and logged in the application.

R3: User must be a taxi driver.

R4: The taxi driver must be available.

R5: the taxi driver must have received a notification for that ride request.

3.3 Scenarios

3.3.1 Occasional user

Mario, an important businessman, has just arrived by train in our city for the first time. His train had technical issues and now he is a bit late for the meeting on the opposite side of our town, so he does not want to take public transports. The best solution is to call a taxi. He opens the browser on his smartphone and, inserting name and surname, sends a taxi request to our system. As soon as possible he receives a notification with the taxi ID number and the waiting time before the taxi arrival. In less than no time he will reach the meeting location.

3.3.2 Habitual user

The meeting of Mario went very well: he started an important collaboration with a local company. He will have to come back often in our town. He found our service very useful, so he decides to register and download the mobile application. After a couple of weeks he has to come back. Unfortunately his train had technical issues, and he risks to be late to the meeting again. This time he has our application installed on his mobile device, so with only few taps and no additional information provided requires a taxi. As soon as possible he receives a notification with the taxi ID number and the waiting time before the taxi arrival. In less than no time he will reach the meeting location.

3.3.3 Taxi reservation

After another couple of weeks, Mario has to come back in our town again. He thinks that he will never be so unlucky to find a defective train for the third consecutive time, so he decides to make a reservation before leaving, in order to find a taxi waiting for him at the train station at his arrival. Using the mobile application, he fills in all the necessary fields, indicating hour and place of the request. Now he can airily taste his journey.

3.3.4 Deleting a taxi reservation

Mario's train has technical issues. Poor Mario, he's so unlucky! He will be very late, so he does not want to let the taxi wait so long. He opens our application and, 30 minutes before the taxi request, he selects the reservation and deletes it. Once arrived he will require a taxi as usual.

3.3.5 Taxi driver

A taxi driver, Luigi, is having a little break, eating his favourite chocolate bar in his taxi. Suddenly his phone rings: a notification has just arrived. But he's having a break, so he does not answer. After 1 minute, the system automatically records a negative answer from the taxi driver. Luigi slowly ends his bar and starts answering some messages on his smartphone. Suddenly his phone rings. He sees the notification, but he's very busy, so he declines the request. Luigi greets his friend on the chat and goes back to work. After a few minutes his phone rings again. A man called Mario is requesting a taxi in his area, right next to the train station. He accepts the request and starts driving towards the location of the customer.

3.3.6 Taxi availability

After a hard morning, Luigi decides that it is the right time to have lunch. He opens our mobile application and sets his status as unavailable: he does not want to be disturbed during such an important moment. After a hearty lunch, he goes back to work, so he picks up his phone, opens our application and sets his status as available. He works hard for the whole afternoon and, after the last passenger, he's so tired that he forgets the status on available. The system sends him 1, 2, 3 notifications of requests, but Luigi has a huge headache and can not hear the phone ringing. After the third request without answer the system automatically sets the driver as unavailable. Finally Luigi will get its deserved peace.

3.4 Use Case Diagram

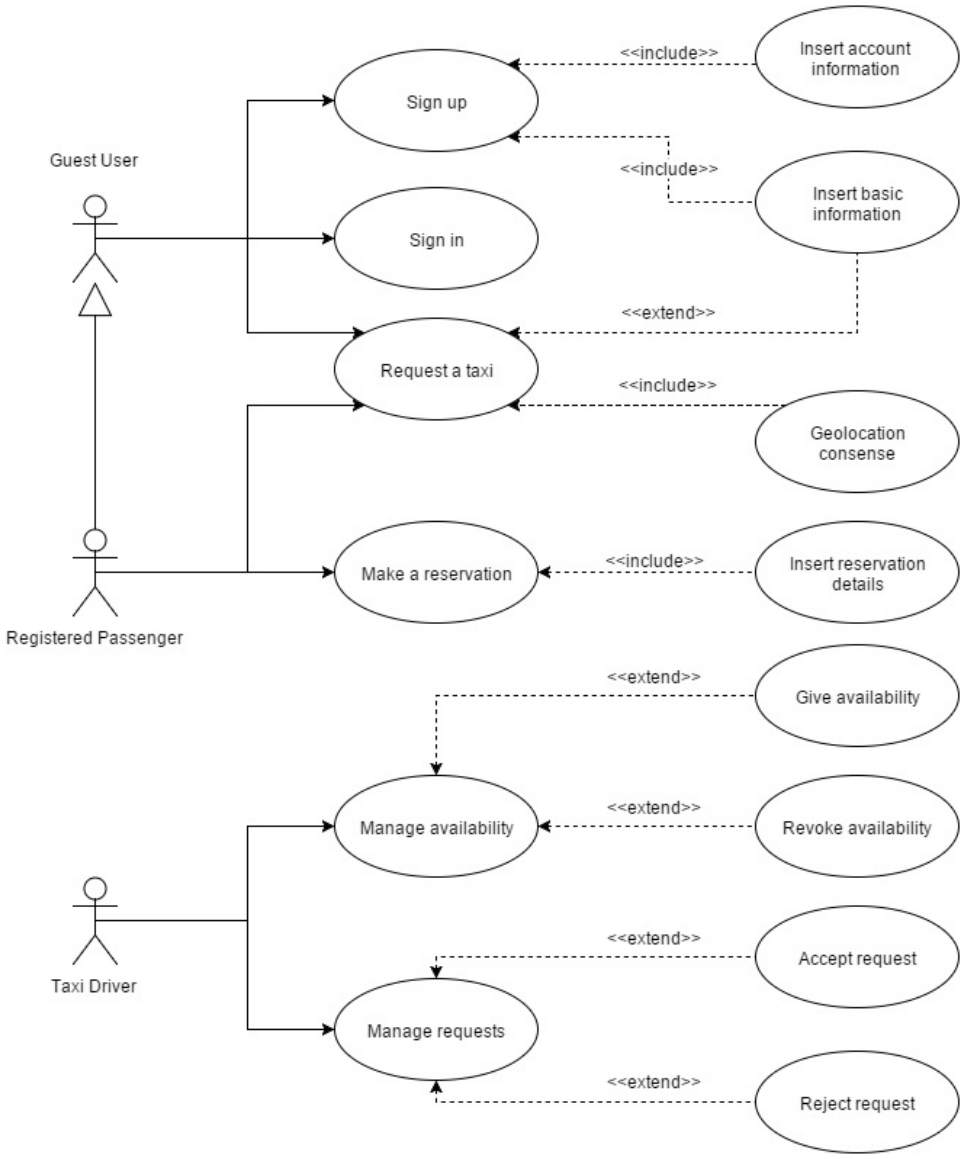

Figure 1: Use Case Diagram for MyTaxiService

3.5 Class Diagram

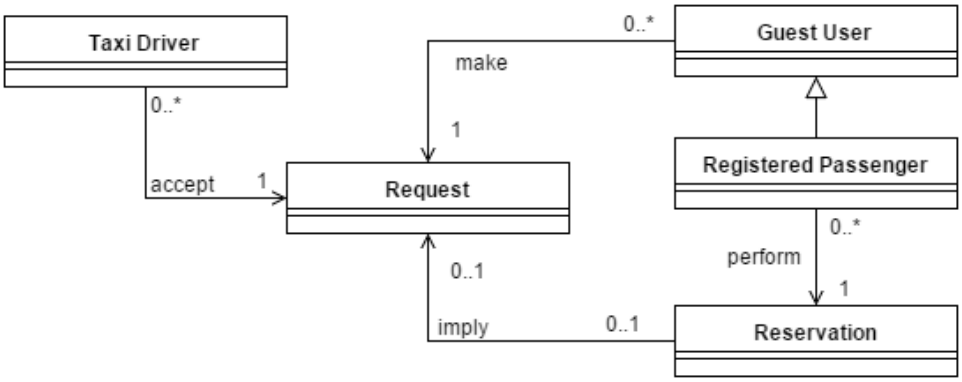

Figure 2: Class Diagram for MyTaxiService