Requirements Analysis and Specifications Document

Figure 1: Politecnico di Milano

Version 0.1

- Claudio Cardinale (mat. 849760)
- Gilles Dejaegere (mat. **MAT**)
- Massimo Dragano (mat. 775392)

Contents

1. Introduction
    1. Description of the given problem
        1. Actual system
    2. Goals
        1. Taxi drivers
        2. Users
    3. Domain properties
    4. Glossary
    5. Assumptions
    6. Constrains
        1. Regulatory policies
        2. Hardware limitations
        3. Interfaces to other applications
        4. Parallel operation
        5. Reference documents
    7. Proposed system
    8. Identifying stakeholders **HERE OR INTO 'Actor identifying'?**
    9. Other considerations about the system
2. Actors identifying
3. Requirements
    1. Functional requirements
        1. Taxi drivers
        2. Users
    2. Non-functional requirements
        1. User interface
        2. Documentation
        3. Architectural consideration
4. Scenario identifying
    1. Scenario 1
    2. Scenario 2
    3. Scenario 3
    4. Scenario 4
    5. Scenario 5
    6. Scenario 6
    7. Scenario 7
5. UML models
    1. Use case diagram

    2. Use case description
    3. Class diagram
    4. Sequence diagram
    5. State diagram
6. Alloy modeling
7. Used tools

Introduction

Description of the given problem

We will project and implement myTaxiService, which is a service based on mobile applications and a web applications, with two different target of people:

- taxi driver
- clients

The system allow clients to reserve taxi via mobile or web app, using GPS position to identify client's zone (but the client can insert it manually) and find taxi in the same zone.
On the other side the mobile app allow to taxi driver to accept or reject a ride request and to communicate automatically his position (so the zone).

The system includes extra services and functionalities such as taxi sharing

Actual system

Until now the taxi company has a system where the clients have to call a call center communicating its position via voice (so it can be not correct), the call center's operator insert the request into an internal information system and the taxi driver can accept or reject it via an dedicated hardware device.

The system send automatically an SMS to the client with the estimated arrival time and the taxi name.

This system store taxi information into a Mysql database.

Goals

The main goal of the system is to be more efficient and reliable than the existing one in order to decrease costs of the taxi management and offer a better service to the users.

Taxi drivers:

- [G1] Allow taxi drivers to sign up into the system.
- [G2] Allow taxi drivers to log in the system.
- [G3] Allow taxi drivers to precise to the system if they are available or not.
- Taxi drivers should receive a notification for incoming request. **is it a goal ?**

- Taxi drivers should receive a notification if they have to take care of another user (during a shared ride). **is it a goal ?**
- [G4] Allow taxi drivers to accept or decline incoming requests for an immediate ride
- [G5] Allow taxi drivers to accept or decline incoming request for a later reservation.

Users:

- [G6] Allow users to request for an immediate taxi ride.
- [G7] Allow users to request for the reservation of a taxi at least two hours in advance.
- Users should receive a notification with the code of the taxi that takes car of the user's request. **is it a goal ?**
- Users should be notified if no taxi driver is able to perform the users request. **Is this a goal ?**
- [G8] Allow users to require to share the taxi.

Domain properties

We suppose that these properties hold in the analyzed world :

- Actual drivers are already registered on the previous system
- A taxi driver accepting a ride of reservation will actually take care of the request.
- A user requiring a taxi will actually take it.
- All the GPS always give the right position.
- The GPS of the taxi drivers can not be switched of.
- Taxi drivers answer all types of demands in less than 5 minutes.
- The user pays the taxi driver directly for each commission.
- A taxi can be on only one zone at the same time and this is the real zone.
- Users make a reservation two hours before of the ride **Here as domain or do we have to do the requirements into G7?**
- When a new taxi driver join in the taxi company the taxi company register him to the information system. Analogously when a taxi driver exit from the company, the company delete him from the information system.
- The taxi arrives at start point with max 30 minutes of delay

Glossary

- User: he is a client of the service. He should insert each time he performs a request the following information

  - Name
  - Phone number
  - Position, it can be taken automatically from GPS (either via APP or Web browser)

- Taxi driver: he is a taxi driver registered on the taxi company, that grants to taxi driver the access to this information system
- Queue: it is the taxi queue, when more than one taxi are in the same zone, there is a FIFO queue. So in this way when there is a new client the oldest taxi can take it. there is a queue for each zone.
- Ride: it starts when the taxi receive the request and ends when leave the last client of the ride. The simple ride is specified by start ride, user and taxi; but other ride types (like reservation or taxi sharing) have other parameters.
- Taxi sharing: it is the possibility that if different people (it's not require that they know each other) of the same start zone go to the same direction, even if the end is not the same, to use the same taxi and to have an unique group fee. A sharing ride is identified by users that use it and for each user the start and end point
- Reservation: it is the ability to reserve a taxi until two hours before time of ride, so when a reservation is done the system make a normal taxi request 10 minutes before the ride. The reservation is identified by start point, end point, user and time.
- Taxi request: it is the request that the system send (automatically or after an user request) to taxi to specify a ride, specifying start point, user and other elements if they are available.
- User request: it is the request for a taxi drive as soon as possible, it contains the the user data and the start point that can be get by GPS (current position) or inserting manually
- Zone: is a zone of approximately 2 km^2, the city is split into these zones. From taxi position the system get his zone and inserts the taxi into the zone queue. So the system guarantees a fair management of taxi queues
- Task: a task is an action done automatically from the server, for example "send request 10 minutes before ride" is a task
- Taxi: is a means of transport that can bring only 4 passengers.
- System: is the the new system that we will create with the database of old system.

Assumptions

- There is an old system as described above.
- There exist an mobile application for users where user can make a reservation using the GPS position or by inserting his position
- The users are not registered on the system, because we need only their

name and their position. **SEE REQUIREMENTS** An user is identified by his personal data: name and phone number
- There are only normal taxis for 4 passengers.
- The registration/deletion by company of a taxi driver is done in the same way of the old system, so we don't have to do this part.
- We need information only about taxi driver, not about taxi vehicle. So we store informations only about taxi driver.

Constrains

Regulatory policies

The system must be require to user/taxi driver the permission to get his position and he have to manage sensible data (position, phone number) respecting the privacy law

Hardware limitations

- Mobile app
  - taxi driver:
    - 3G connection
    - GPS
    - Space for app package
  - user:
    - 3G connection
    - Space for app package
- Web app
  - Modern browser with AJAX support
  - ...

Interfaces to other applications

Interface with the old system. The new system will interface with the Mysql database of the old system.

Parallel operation

The server support parallel operations from different users and different taxi drivers.

Reference documents

...

Proposed system

We will implement a client-server architecture (Fig. 2) based on common REST API, so with just one server application we manage web application and mobile application, obviously we will have version for taxi driver and version for users.

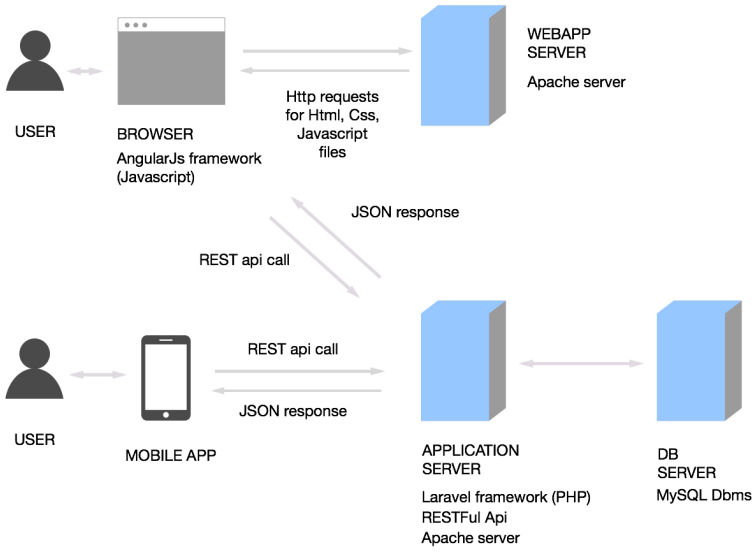

Figure 2: Architecture

WRITE MORE DETAILS

Identifying stakeholders HERE OR INTO 'Actor identifying'?

Do we have to write city or teachers?

Other considerations about the system

Actors identifying

The actors of our system are basically two:

- Taxi driver: is a taxi driver registered automatically on the system by the taxi company
- User: he doesn't need to register himself to the system, since he uses the system only to call a taxi (so he have to insert only basic personal information and location)

Requirements

Functional requirements

Assuming that the domain properties stipulated in the paragraph [1.3] hold, and, in order to fulfil the goals listed in paragraph [1.2], the following requirements can be derived.

The requirements are grouped under each goal from which it is derived. The goals are grouped following under the users concerned.

Taxi drivers:

- [G1] Allow taxi drivers to sign up into the system:
  - The system must be able to check if it is an official taxi driver.
  - The system only allows official taxi drivers to register. **SEE DOMAIN, probably we should remove this**
- [G2] Allow taxi drivers to log in the system:
  - The system must be able to check if the password provided is correct.
  - The system must only let the taxi driver log in if the provided password is correct.
- [G3] Allow taxi drivers to precise to the system if they are available or not:
  - The system must put the taxi driver in the appropriate queue when the taxi user becomes available.
  - The system must remove the taxi driver from the appropriate queue if he becomes unavailable.
- Taxi drivers should receive a notification for incoming request. **is it a goal ?**
- Taxi drivers should receive a notification if they have to take care of another user (during a shared ride). **is it a goal ?**
- [G4] Allow taxi drivers to accept or decline incoming requests for an immediate ride:
  - The system must ask to the taxi driver if he accepts to perform the ride.
  - The system must replace a taxi driver at the end of the queue if he declines the ride.
  - The system must ask to the next taxi driver if the former one declined the ride.
  - The system must notify the user with the code of the taxi driver who has accepted the ride.

  - The system must notify the user if no taxi driver in the queue accept the ride.

- [G5] Allow taxi drivers to accept or decline incoming request for a later reservation:
  - The system must ask to the taxi driver if he accepts to perform the reservation.
  - The system must ask to the next taxi driver if the former one declined the reservation.
  - The system must notify the user with the code of the taxi driver who has accepted the reservation.
  - The system must notify the user if no taxi driver in the queue accept the reservation.
  -

Users:

- [G6] Allow users to request for an immediate taxi ride :
  - The system must be able to check the position of the user.
  - The system must not accept request of users outside the area of the city.
  - The system must transfer the request to the appropriate taxi driver.
- [G7] Allow users to request for the reservation of a taxi at least two hours in advance.
  - The system must be able to check to origin and the destination of reservation.
  - The system must not accept reservations with an origin outside the area of the city.
  - The system must transfer the reservation to the appropriate taxi driver.
- Users should receive a notification with the code of the taxi that takes car of the user's request. **is it a goal ?**
- Users should be notified if no taxi driver is able to perform the users request. **Is this a goal ?**
- [G8] Allow users to require to share the taxi.
  - The system must be able to find if there are reservations or request for the same time period and having corresponding journeys.

Non-functional requirements

User interface

Documentation

We will draft these documents to well-organize our work in the way to do in a fewer time the best work as possible:

- RASD: Requirement Analysis and Specification Document, to well- understand the given problem and to analyze in a detailed way which are our goals and how to reach them defining requirements and specification.
- DD: Design Document, to define the real structure of our web application and its tiers.
- Testing Document: a report of our testing experience of another myTaxyService project.

Architectural consideration

We will use the following technologies:

- Apache with php (with laravel framework) as API server and task service
- Mysql as sql server to store data persistently, it is the same of the old system
- Apache server for static documents
- RESTFull and JSON for API communication over HTTP(S)
- Javascript (with angularJs framework), CSS and HTM to create responsive site that communicate to server using REST API. These files are got via HTTP(S)
- Modern browser with javascript and ajax support
- Java and swing respectively for android and iOS apps, using original SDK
- Internet connection to communications of data

Scenario identifying

Here some possible scenarios of use of this application

Scenario 1

John wants to go to home after the office and he wants to do that as soon as. So in the morning (so he respects the constrains of two hours before) he reserve a taxi at the same time of the end of his job for a ride that start from his office and ends on his home.
When he goes out from office he finds the taxi on the street that bring him to his home.

Scenario 2

Some friends live in the same zones and want to go to airport for a trip together, they want a cheap solution. So they chose taxi sharing option to go to airport. The morning of the trip's day all friends request a taxi with sharing option.
Since they are 6 and a taxi can bring only 4 passengers they need at least 2 taxis, so 4 friends are in the same taxi while two others are in other two taxis, each of them filled by other people that have chosen the taxi sharing option and start from the same zone and have to go in the same direction.

Scenario 3

Scenario 4

Scenario 5

Scenario 6

Scenario 7

Uml models

Use case diagram

Use case description

Class diagram

Sequence diagram

State diagram

Alloy modeling

Used tools

The tools we used to create this RASD document are:

- DIA: for uml models
- Pencil: for mockup
- Gedit and ReText: to write MarkDown with spell check
- Pandoc: to create pdf