Politecnico di Milano

Scuola di Ingegneria Industriale e dell'Informazione

M.Sc. in Computer Science and Engineering

---

myTaxiService

Software Design Document

Authors:

Angelo Gallarello

Edoardo Longo

Giacomo Locci

November 23, 2015

Contents

1 Overview

myTaxiService is a taxi service that will operate in a big city; the main purpose is to simplify the access of passengers to the service and to guarantee a fair management of the taxi queues.

The main stakeholders of the system are the *Users*, the *Taxi Drivers* and the *Operators* as highlighted in *section 1.3* of the *RASD*.

The system is composed of four main core applications :

- Mobile Application (User)

- Web Application

- Mobile Application (Taxi Driver)

- Back-End Application

as stated in *section 1.2.* of the *RASD* It's important to highlight that in this document the design of mobile application is based on the Android platform.

2 High Level Components

The system could be divide in three main high level components that do not necessarily correspond only to one real application:

Server

The Server component is the kernel of the service we want to provide, it incorporates most of the *business logic*, it stores most of the *data* and it provides programmatic interfaces to the clients.

User Client

The User Client components is an high level representation of the real clients available to the users of our service. It's modeled as a *thin client* and it relies on the *Server* to fulfill its tasks.

Taxi Driver Client

The Taxi Driver Client component is an high level representation of the real clients available to the taxi drivers registered to the service. It's modeled as a *thin client* and it relies on the *Server* to fulfill its tasks.

2.0.1 Components Interaction

From a high level perspective the system is design following the well known *client-server* paradigm.
The interaction between the components is handled by the Server that provides a programmatic interface that is able to receive remote call from the clients.
The clients never communicate directly with one another.

3 Component View

This section highlights the main features and roles of every component of the system. Moreover it describes the internal interfaces between different classes of every component.
External interfaces between components are described in *section 4*

3.1 Server

The Server is composed of:

Back-End Application

As stated in *section 1.2.2* of the *RASD*, the *Back-End Application* is the system component that handles most of the business logic.
The application is written in *Java EE* and to fulfill its tasks (see *section 3.5.3* of the *RASD*) it needs to interface with the Internet network using the *HTTPS protocol* and the *JAVA API for RESTful Web Service*[1], with a *MySQL database* and with external Google Maps API.

---

[1]See https://jax-rs-spec.java.net/

Back-End Internal Interfaces

The *Back-End Application* is built to be very modular and to grant *interchangeability* between components.

There are four main classes that constitute the kernel of the application :

- **QueueManager** Handles queue policies.

- **RideManager** Creates and manage rides. Is connected to the *RequestManager* via the *RideManagerInterface* and directly depends on *QueueManager*

- **ActorManager** Create and update data about users an taxi drivers. Is connected to the *RequestManager* via the *ActorManagerInterface*

- **PositionManager** Update taxi drivers position. Is connected to the *RequestManager* via the *PositionManagerInterface* and directly depends on *QueueManager*

- **RequestManager** Get and build *Request* object from the requests received via *HTTP*

MySQL Database

The MySQL database fulfill the task off storing and granting access to all the data generated and used by the service.
A *database dump* is performed daily during the period of minor activity of the service [2].
The connection between the *Java EE* application and the databased is supported by the *JDBC connector*[3]

3.2 User Client

Different real clients are available to the end users of the system.
As stated in *section 1.2.2* of the *RASD* a native mobile application is developed for Android, iOS, Blackberry and WP.
Moreover a Web Application is also available.
To fulfill the requirements expressed in *section 3.5.1* and *section 3.5.2* of the *RASD*, all the clients need to communicate with the Server making calls to the REST API using platform specific API for REST HTTP calls.

---

[2]At first, when no activity data is available, the dump will be performed at 04:00 A.M
[3]See `http://dev.mysql.com/downloads/connector/j/`

3.3 Taxi Driver Client

Different real clients are available to the taxi driver registered to *myTaxiService.*
As stated in *section 1.2.2* of the *RASD* a native mobile application is developed for Android, iOS, Blackberry and WP.
To fulfill the requirements expressed in *section 3.5.1* and *section 3.5.2* of the *RASD*, all the clients need to communicate with the Server making calls to the REST API using platform specific API for REST HTTP calls.

3.4 Clients Internal Interfaces

Mobile clients are composed mainly by subclass of platform specific components.
Interfaces between components are therefore specified in the SDK of each platform. However it's important to highlight that every mobile application has to interface with a *Network Component* that handles *HTTP* requests.

4 Components Interfaces

This section provides a description of the interfaces between the main components of the system.
Internal interfaces between different objects of every component are described in *section 3*

4.1 Back-End Application - Database