# Change Log

# Preface

# Table of Contents

# List of Tables

# List of Illustrations

1. Introduction

Along this section we make on overview of the present document. We treat aspects like Purpose, Scope and Overview. Glossary and references are also included in this section.

* 1. Purpose

The present is the **R**equirement **A**nalysis and **S**pecification **D**ocument, RASD, concerning the project MyTaxiService for the Software Engineering 2 course at Politecnico di Milano.

The purpose of this document is to present a complete description of the product and the analysis of its domain (which includes exposal of stakeholders, scenarios, use cases, constraints and assumptions), in order to join them together to obtain the corresponding software requirements. Therefore, we also provide here a technical sheet for the further development, and an artifact which might even be used as a contract between the eventual developers and customer(s).

* 1. Scope

MyTaxiService is a Milano’s government proposal for optimizing its taxi service, by simplifying the access of the passengers to the services and guaranteeing a fair management of taxi queues.

The passengers will be able to make requests for taxi services either through the MyTaxiServices’s web site or its mobile application, by sending the corresponding *request for service information*. The system then replies to the passenger with the *accepted request information*, and he is successfully served. The passengers can also reserve taxi services in advance and share the taxi with other passengers.

The taxi drivers will be able to receive requests for taxi services in the mobile application only, whenever they have informed the system about their availability. A received request is accompanied by its corresponding *incoming request information*. In the moment that the request is accepted by the driver, the passenger is informed and receives the *accepted request information*.

The requests are managed and assigned to available drivers, according to the position provided by their taxi’s GPS. The city is divided in zones and each one of these has an associated queue of available taxis. The request is assigned to the first driver in the corresponding queue.

* 1. Definitions, acronyms, and abbreviations

Accepted request information: corresponds to the following information: taxi’s code, estimated arrival time, fee to be paid to the taxi driver, and possibly how many people the car will be shared with. It is received by the passenger when his request has been accepted.

Compatible request:

Incoming request information: corresponds to the following information: the origin(s), destination(s), the eventually payed fee for the trip, and possibly the amount of passengers. It is received by the taxi driver together with a request for a service.

Process request: the processing of a request means to create the *incoming request information*. It can happen in either two ways, depending on the *request for service information* received:

* The passenger does not want to share the taxi, so the origin, destination and amount of people are set according to the information provided by the user, and the fee is calculated according to this positions.
* The passenger wants to share the taxi, so the system tries to find up to three compatible requests among the received ones within the established *searching time*. If it succeeds, then the origins, destinations, fee and amount of passengers are set accordingly. Else, the request is processed as if the passenger did not want to share the taxi.

This moment of this processing depends on the specified pick-up time. If it is *now*, it takes place immediately; else, it takes place then minutes before the specified time. The calculation of the zone and the fee are also performed in this procedure.

Recognize request: the recognition of a request means to identify the *request for service information* just received in order to know the type of the request (normal, sharing, or reservation), and support the subsequent processing.

Request for service information: corresponds to the following information: the origin and destination of the trip, the amount of people, whether he wants to share or not the taxi, and the time he wants to be picked up (which can be *now* or a moment no before than 2 hours later). It is provided by the passenger when he makes a request for a service. This is considered to be correct if both origin and destination are places in Milano. This is considered to be complete if all fields have been filled in.

Process a taxi driver's answer: In the case of a positive answer the system create and send accepted requests to all the passengers concerning by the incoming request that induced this answer. In the case of a negative answer, the system fairly chose an other taxi driver to who send the incoming request that induced this answer.

Searching time:

* 1. References
  2. Overview

1. Overall description

The intention of this section is to contextualize the developing software product and provide the background that justifies the subsequent definition of the requirements. We start by putting the MyTaxiService product in perspective with respect to its surrounding world, to proceed next to define its functionalities in terms of goals. After that, deeper analysis to relevant aspects are presented; those aspects include Stakeholders, users and actors, Scenarios, Constraints, and Assumptions and dependencies. The results are formalized through Use case models and a Class diagram.

* 1. Product perspective

In this subsection we present the MyTaxiService software from a context-oriented perspective. We start by applying “The world and the machine” paradigm to this particular case, and then we describe the interfaces that will allow the application satisfy the requirements.

MyTaxiService makes neither part of any already existing system nor replace an existing one, so there are no several constraints on the required interfaces.

* + 1. The world and the machine

The phenomena we describe here (both of the world and machine) will be listed in the form of events.

This first list corresponds to world elements that are not observable by the machine but which are somehow relevant. Since they are not observable, we only name a small group of them:

* A taxi driver picks up a passenger
* A taxi driver takes a passenger to his destination
* A taxi driver gets paid for his service
* A taxi driver is stuck in traffic
* A street is in bad conditions
* A taxi is in bad conditions

The list below shows the world phenomena that are observable by the machine:

* A passenger requests a taxi service (normal, shared or reservation)
* A taxi driver informs his availability
* A taxi driver accepts a passenger’s request
* A taxi driver declines a passenger’s request
* A passenger cancels an accepted request
* A taxi driver cancels an accepted request

The third list exposes the events that occur inside the machine but are still observable by the world:

* A processed request is sent to a taxi driver
* A passenger is informed about his request’s result
* A taxi driver is informed about the request cancelation
* A passenger is informed about the request cancelation

Finally we have the phenomena occurring inside the machine and unreachable by the world. We list only few of them since they refer to the internal work of the machine and it does not concern the requirements process:

* A request is processed (the address is assigned to a zone, available taxi driver is found, trip-mates are searched, etc.)
* A request is recognized

WORLD

### External systems interfaces

The following table relates the external systems that will interact with MyTaxiService, the nature of the interface and the description of the interaction.

|  |  |  |
| --- | --- | --- |
| External system | Nature of interface | Description of interaction |
| Taxi’s GPS | Provided | MyTaxiService will be able to know the current position of each taxi through their installed GPS. This will be useful to detect the zone in which the driver is located and assign him the corresponding requests. |
| Maps server | Provided | MyTaxiService will use the maps service to perform three tasks:   * Resolve the passenger provided address to actual coordinates * Assign the driver to a zone based on his GPS position * Display a map to both the driver and passenger with the driver’s and the pick-up position |
| Milano government’s information system | Provided | MyTaxiService will access to a database of the government of Milano to obtain the valid assigned codes and verify the driver’s indentity. |

Table 1: External system interfaces

### User interfaces

MyTaxiService will interact with two types of users: passengers and taxi drivers.

A passenger can interact with the system through the web site or the mobile application, and the system must allow him to perform all the following logical actions regardless where he accesses from:

* Make a request for a taxi service (normal, shared or reservation)
* View the current state of his request
* View the response to his request
* View the information of the taxi driver
* View the driver’s and the pick-up position
* Cancel an accepted request

A taxi driver can interact with the system only through the mobile application, and the system must allow them to perform all the following logical actions:

* Inform his availability
* View received requests
* Accept or decline received requests
* View his own position and the pick-up point’s position
* Cancel an accepted request

We do not include here any description of the actual graphic user interface since it does not concern the requirements process.

According to one of the project’s constraints, the system will have to provide an interface to eventual developers to allow them to implement further functionalities. Only the following basic primitives will be exposed:

### Operations

According to the previous section, MyTaxiService will support access only to two type of users. The interaction that they will hold with the system shall only be under one mode of operation, so no additional ones will be implemented (e. g. administrator mode). The description of such operation mode has already been described.

* 1. Product functions – Goals

In this subsection we expose the product functions in the form of goals. Since we are following “*The world and the machine”* approach, such goals are listed in the form of events whose accomplishment shall be supported by the system. Each goal has a dedicated section that makes a corresponding brief description.

### G1: Passenger can request a taxi either through a web application or a mobile application

### G2: Taxi driver informs the system about his/her availability only through a mobile application

### G3: Taxi driver may confirm that he/she is taking care of a certain received request only through a mobile application

With this goal we want the system to support the taxi driver in receiving requests and the subsequent possible actions:

* Accept the request
* Decline the request
* Cancel accepted request
* View current position, origin(s) and destination(s)

### G4: Requests for taxi services are fairly managed

With this goal we want the system to manage and assign the received requests in a fair manner through the use of zone queues and specific policies:

* Requests are sent to the first driver in the queue
* If a driver accepts a request he is removed from the queue
* If a driver declines a request he is sent to the bottom of the queue
* If a driver cancels an already accepted request he is sent to the bottom of the queue
* If a taxi driver receives a cancelation he is sent to the top of the queue
* If a driver enters in a new zone he is sent to the bottom of the queue

### G5: Passengers can enable a taxi sharing option

### G6: Passengers can reserve taxi service in advance

* 1. Stakeholders, users and actors

This subsection begins with a briefly description of the people who are interested in the development of the MyTaxiService product. Further information about those which will interact closer with the system is provided afterwards.

* + 1. Stakeholders

The following table presents the stakeholders concerning the MyTaxiService software creation.

|  |  |
| --- | --- |
| Stakeholder | Description |
| Government of Milano | The project is initiated by the government of the city of Milano, which we assume is being represented by the course teacher. |
| Taxi drivers in Milano |  |
| Taxi users in Milano |  |
| Production engineers | The project team that will design the solution and plan its realization. Represented by the authors of this document. |
| Developers | It will be an external team, actually we will use the source code of an already existing project. |
| Testers | Our project team will test the solution, identify bugs and inspect the code as well |
| Communicators | Our project team will provide some reviews during the evolution of the project |

Table 2: Stakeholders

* + 1. Users and actors

The description of the product allows us to have almost the same entities as users and as actors. The following table presents those entities and their corresponding role.

|  |  |  |
| --- | --- | --- |
| Entity | Description | Role |
| Passenger | Person who makes use of the MyTaxiService to make a request for a taxi service. He needs to have registered and to log in in order to send a request. He can send requests either through the web site or the mobile application. | User, Actor. |
| Taxi driver | Person who makes use of the MyTaxiService to attend requests for taxi services. He must have an account to log-in into the system, which includes a taxi code. He receives the requests in the mobile application. | User, Actor. |
| MyTaxiService Developer | Developer who uses the programmatic interfaces of the solution to add some features on top of it. | User. |

Table 3: Users and actors

* 1. Scenarios

In this section we mean to give an initial description of the expected system behavior by presenting informal but concrete examples in the form of scenarios.

### Scenario 1: A passenger is successfully served

### Scenario 2: A taxi driver declines the request

Mario celebrated one of his friend's birthday in the downtown, and after that he wanted to go home and decided to call a taxi using the mobile application of MyTaxiService. He opened the application and filled the following data:

* The location from where he wants to be picked up: his current location
* The destination location: his home address
* When he wants this ride: now
* For how many people he orders the ride: 1
* If he wants to share the ride: no

The application asked the taxi driver 1256, who was the first in the queue related to the area from where the ride is supposed to start, if he wanted to take care of this ride. Unfortunately, the taxi driver didn't want to take care of this ride, he send a negative answer to the application. Then, the application put the taxi driver 1256 at the end of the queue and asked the taxi driver 230, who was the second in the queue, for the same request. This time, the taxi driver accepted the request. Thus, the application changed the status of this taxi driver to unavailable and put him out of the queue.

Finally, Mario received a positive answer from the application with the number of the taxi he will take. The taxi 230 came to pick him around 10 minutes later and he went home.

### Scenario 3: A passenger makes a reservation

### Scenario 4: A passenger makes a sharing request

At 5 p. m. in the afternoon, Fabio is so tired of his bad luck day. The previous night he stayed up all night long studying for his Software Engineering 2 final exam, so he did not set the alarm and he missed the review lesson. Later, in the hurry for getting to the next class on time, he spilled his coffee and forgot the money for lunch. Expecting to get back home soon, Fabio makes a request for a taxi trough the MyTaxiService app, and after filling out his origin and destination he enables the trip-sharing option to save some money. The confirmation message informs that someone has joined his request, and the day totally changed for Fabio when he got into the car and realized that his trip-mate was his Software Engineering teacher. He got an excellent score on his exam, and MyTaxiService got a 5-star service evaluation.

* 1. Use Cases

The previous informal scenarios are now formalized by means of use cases. Initially, all the use cases are listed through the Use case model, and then they are described individually.

### Request taxi service

### Cancel accepted request (by the passenger)

### Cancel accepted request (by the taxi driver)

### Create account

### Authenticate

* 1. Constraints

In this subsection we describe the constraints that the software product must be developed under.

* **Interfaces to other applications:** MyTaxiService needs to deal with a geolocation application to locate both taxi drivers and users, and it needs to deal with a map application as well in order to determine itineraries.
* **Regulatory policies:** MyTaxiService needs to respect the taxi economical and legal frame, such as ride prices for day, night, week-end or even legal duration taxi drivers are allowed to work consecutively.
  1. Class model

The class model presented below is intended to provide a static view of the interactions that the real-world entities will hold between them, once the MyTaxiService is implemented and deployed. Brief descriptions of such entities and relationships are also given.

* 1. Assumptions and dependencies

Not login.

Radius to pick up and destination.

1. Specific requirements

This section presents the specific requirements for the MyTaxiSerive software product, which are grouped according to the goal that each one contributes to achieve. Some requirements appear in more than one goal but they are uniquely identified. The identifiers are then separated by functional and nonfunctional.

## G1: Passenger can request a taxi either through a web application or a mobile application

R1: The system must be accessible by the passengers through the website and the mobile applications

R2: The system must provide a form in order to allow passengers to make a request (refer to 1.3 section, request for service information)

R3: The system must be able to process the request

R4: The system must be able to send a message to a taxi driver with the incoming request informations

R5:

## G2: Taxi driver informs the system about his/her availability

## G3: Taxi driver may confirm that he/she is taking care of a certain received request only through a mobile application

R: The system must be accessible by the taxi drivers through the mobile application only

R: The system must allow taxi drivers to create an account

R: The system must allow registered taxi drivers to login

R: The system must allow taxi drivers to inform their availability

R: The system must send a processed request only to the first available taxi driver in the zone queue that corresponds to the origin of the request

R: When a taxi driver receives an incoming request, the system must allow him to see the *incoming request information*

R: The system must allow taxi drivers to accept an incoming request

R: When a taxi driver accepts an incoming request, the system must change is status to not available

R: When a taxi driver accepts an incoming request, the system must send to the passenger the *accepted request information*

R: The system must allow taxi drivers to decline an incoming request

R: When a taxi driver declines an incoming request, the system must inform it to the passenger

R: The system must allow taxi drivers to cancel an already accepted request

R: When a taxi driver cancels an already accepted request, the system must inform it to the passenger

R: When a taxi driver has accepted an incoming request, the system must allow him to see in a map his current location and the passenger(s) origin(s) and destination(s)

## G4: Requests for taxi services are fairly managed

R: The system must be able to recognize the requests with correct *request for service information*

R: The system must be able to process the recognized requests

R: The system must use zone queues to manage available taxi drivers

R: The system must send a processed request only to the first available taxi driver in the zone queue that corresponds to the origin of the request

R: The system must use the taxi’s GPS to know the position of the driver

R: The system must use the Map service to assign a zone queue to the driver according to its position

R: When the system has identified the zone of a just available taxi driver, it must send him to the bottom of the zone queue

R: The system must refresh the zone of every available taxi driver every 5 minutes

R: When an available taxi driver changes his zone, the system must send him to the bottom of the zone queue

R: The system must send to the bottom of the zone queue those taxi drivers who decline a received request

R: The system must send to the bottom of the zone queue those taxi drivers who cancel an already accepted request

R: The system must send to the top of the zone queue those taxi drivers that have received a cancelation for an already accepted request

R: The system must remove from of the zone queue those taxi drivers that change their status to not available

## G5: Passengers can enable a taxi sharing option

## G6: Passengers can reserve taxi service in advance

1. Alloy Modeling
   1. Entities
   2. Facts
   3. Predicates
   4. Functions
   5. Asserts
   6. Generated world
2. Appendix
   1. Used software
   2. Worked hours
   3. Revisions