



**POLITECNICO**  
MILANO 1863

# **Travlendar+ project**

## **Requirement Analysis and Specification Document**

RICCARDO FACCHINI

ANDREA GUGLIELMETTI

October 13, 2017

## Deliverable specific information

---

<b>Deliverable:</b>	RASD
<b>Title:</b>	Requirement Analysis and Verification Document
<b>Authors:</b>	Riccardo Facchini - Andrea Guglielmetti
<b>Version:</b>	0.1
<b>Date:</b>	October 13, 2017
<b>Download page:</b>	<a href="https://github.com/Riccardo95Facchini/FacchiniGuglielmetti.git">https://github.com/Riccardo95Facchini/FacchiniGuglielmetti.git</a>
<b>Copyright:</b>	Copyright © 2017, Riccardo Facchini - Andrea Guglielmetti – All rights reserved

---

## Contents

<b>Deliverable specific information</b>	<b>1</b>
<b>Table of Contents</b>	<b>2</b>
<b>List of Figures</b>	<b>3</b>
<b>List of Tables</b>	<b>4</b>
<b>1 Introduction</b>	<b>5</b>
1.1 Purpose	5
1.2 Scope	5
1.3 Stakeholders	5
1.4 Definitions, acronyms and abbreviations	6
1.4.1 Definitions	6
1.4.2 Acronyms	6
1.4.3 Abbreviations	6
1.4.4 Revision History	6
1.5 Reference documents	6
<b>2 Overall Description</b>	<b>7</b>
2.1 Product Perspective	7
2.2 Product Functions	9
2.3 User Characteristics	9
2.4 Domain Assumptions	9
<b>3 Specific Requirements</b>	<b>10</b>
3.1 External Interface Requirements	10
3.1.1 User Interfaces	10
3.1.2 Hardware Interfaces	10
3.1.3 Software Interfaces	11
3.1.4 Communication Interfaces	11
3.2 Functional Requirements	12
3.2.1 Registration	12
3.3 Performance Requirements	16
3.4 Design Constraints	17
3.4.1 Standards Compliance	17
3.4.2 Hardware Limitations	17
3.5 Software System Attributes	18
3.5.1 Reliability	18
3.5.2 Availability	18
3.5.3 Safety and Privacy Constraints	18
3.5.4 Maintainability	18
3.5.5 Portability	18

## List of Figures

1	Class Diagram . . . . .	8
2	<i>Registration</i> activity diagram . . . . .	14
3	<i>Registration</i> sequence diagram . . . . .	15

## List of Tables

1	<i>Registration</i> use case description . . . . .	13
---	--	----

## 1 Introduction

### 1.1 Purpose

This Requirement Analysis and Specification Document (RASD for short) document has the purpose of fully describing to a wide range of potential readers the system and to function as a base for legal agreements between developers and other parties.

In the following pages there will be precise descriptions of all the functional and non-functional requirements, the different scenarios and cases of interaction between the system and the users, with attention to what the users need from it, the domain of the system and the constraints that it implies.

The readers of this document comprise the developers of the system and its applications and agents from the local public transportation agencies or independent company in similar professions such as taxi businesses or bike/car sharing companies.

### 1.2 Scope

The scope of this project is to develop a system called Travlendar+ that will allow in the most possible efficient way the paring of daily commutes and the management of scheduled appointments and meetings, by providing for each situation the best alternatives of moving throughout the city both for work related reasons and for personal motives.

Users of Travlendar+ can create a calendar with every appointment paired with time and place, the system will then compute the best way of reaching each location in time by choosing between every commuting option available at the moment and taking into account the preferences expressed by the user in the customization settings, possible strikes of the local transportation services and the weather, if the location is too far and cannot be reached in time a warning is going to pop up on the screen of the user.

Each time a scheduled appointment is coming up a notification is going to appear in advance by a configurable amount of time, the user will then be able to confirm, change or refuse the proposed trip.

### 1.3 Stakeholders

Here are listed all the potential stakeholders with a brief description and how they may be affected by the system:

- **User:** All the individuals that will use the system to schedule their daily commute.
- **Public transportation companies:** Local and international companies that handle public transportation may have an advantage in giving an easy way to integrate their schedules with Travlendar+ as it could mean a higher number of sold tickets.
- **Taxi and Car/Bike sharing companies:** Given that is not always possible for each type of user to walk long distances and public transport does not reach every possible destination they may be interested in a partnership between their service and the system.
- **Mobile network carriers:** They have an interest in providing a network and contracts to connect devices to the service.

## 1.4 Definitions, acronyms and abbreviations

What follows is the list of all the main terms and abbreviations used in the document.

### 1.4.1 Definitions

- **User:** who is using the system to schedule their calendar.
- **Trip:** the plan to move from point A to point B done using one or more means.
- **Travel:** synonymous of trip.
- **System:** All the software needed to deliver all the functionalities desired, often used as a synonymous of Travlendar+.

### 1.4.2 Acronyms

- **RASD:** Requirement Analysis and Specification Document
- **SRS:** Software Requirement Specification. Synonymous of RASD
- **ETA:** Estimated Time of Arrival
- **GPS:** Global Positioning System
- **W3C:** World Wide Web Consortium
- **HTTP:** HyperText Transfer Protocol
- **HTTPS:** HyperText Transfer Protocol over Secure Socket Layer
- **SDK:** Software Development Kit
- **TCP:** Transfer Control Protocol
- **API:** Application Programming Interface
- **RAM:** Random Access Memory
- **UMTS:** Universal Mobile Telecommunications System

### 1.4.3 Abbreviations

No other abbreviations aside from acronyms were used.

### 1.4.4 Revision History

- 07/10/2017 Introduction.
- 09/10/2017 Product perspective and Specific requirements.
- 11/10/2017 Minor fixes to the structure, added subsections 3.5 and expanded Product Perspective.

## 1.5 Reference documents

- Document of the assignment: Mandatory Project Assignments.pdf

## 2 Overall Description

### 2.1 Product Perspective

The system will have three main parts:

1. Mobile application version for phones and tablets.
2. Web browser version.
3. Backend structure to support the functioning of the service.

While the backend structure is needed for the functioning of the service provided, only one of the two applications is needed to interact with the system.

APIs for each component must be provided in order to allow future development and introduction of new functionalities like an automated system for buying of public transportation vehicles.

### Class Diagram

In **Figure 1** is represented the class diagram for the main components of the system-to-be

### Difference between Machine and World

The following is a distinction between the **Machine** and the **World**.

1. Machine: the Machine is the *Product-to-be* (often also referred to as *System* for short).  
The *Machine Domain* on the other hand is everything that the Machine can operate onto, meaning that it can manipulate it or more in general it can control.
2. World: the World is the physical world that interacts with the Machine or that it can be observed by it, it's the environment in which the System will gather information and will be affected by the actions performed.

Machine and World are connected by *Shared Phenomena*, the set of events of the World that are observable by the Machine and the ones the Machine can directly cause with its actions.

An example of a Shared Phenomena may be something as mundane as the rain, since it's clearly an event that happens in the World and at the same time is observable by the Machine via a forecast or a weather report.



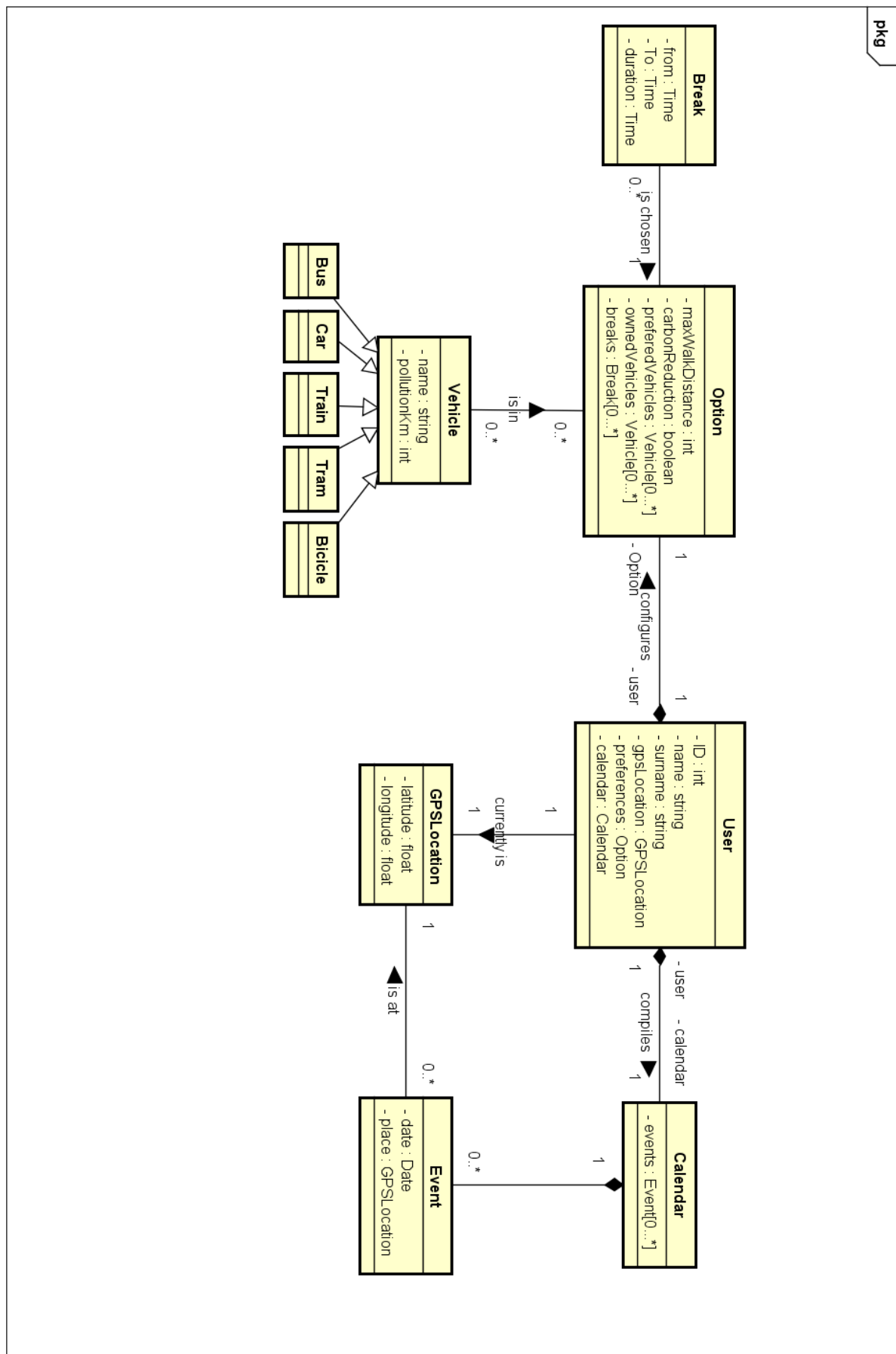


Figure 1: Class Diagram

## 2.2 Product Functions

The system allows each user to create their personal calendar by specifying place and time of each appointment and then view the proposed solution, to be more specific the user can:

1. Register to the system with username and password.
2. Logging to the service.
3. Manage the information of an account and delete it.
4. Specify means of travel preferences.
5. Create an appointment in the calendar.
6. Manage information about appointments and delete them.
7. Schedule flexible lunch/breaks of specific length in a given interval.

## 2.3 User Characteristics

The users interested in using the system should be at least familiar with the concept of navigating a web page and using a smartphone in the day to day routine without needing any technical competence regarding the topic, they must be aware of the laws regarding the public circulation on the streets of the country they wish to use the services provided and need a valid driver licence if they want to use a car and they must possess an electronic payment method to use third party car/bike sharing options.

## 2.4 Domain Assumptions

We assume that the following statements are true:

1. It exists a combination of means of transportation that will take the user to his/her destination even if not before the desired time.
2. The GPS location of the user is always correct.
3. Weather forecasts have a 100% accuracy.
4. Public transportations always arrive on time.
5. Internet connection is never lost.
6. Is possible to integrate the system with already existing application from third party companies and public transportations.

## 3 Specific Requirements

### 3.1 External Interface Requirements

#### 3.1.1 User Interfaces

The main way the users can interact with the system is via the mobile application for their smartphone, the interface should be user-friendly and in particular easy to read, with large and high contrast text to minimize reading problems in direct sunlight; the second way of connecting to the services provided is to use the web application their personal computer. In both cases the user interfaces must satisfy the following constraints:

- The first page must always ask the user to login or register to service.
- After the login, the system redirects the user to his/her home page.
- (Web) A toolbar must be present in every page, except login and registration page.
- (Mobile) A toolbar must be present in the homepage.
- The *Create a meeting* page must provide a guided process to set-up a meeting and clearly show if the created meeting is not reachable in time from the location of a previous appointment.
- The *Manage meetings* page must show a list of user's meetings divided by day and hour, and allows the user to select a meeting to obtain further information.
- The interface must offer the possibility to choose between a set of different languages.
- The user interface must dynamically adapt to the screen size.
- The Mobile and Web application must use the same graphical elements.

In addition to these constraints, other platform-dependent constraints are provided:

- Web Application:  
All the pages must submit to W3C standards.
- Mobile Application:  
All mobile versions must follow the design guidelines provided by the respective platform manufacturer (Android, iOS, Windows ...).

#### 3.1.2 Hardware Interfaces

The web application needs any personal computer connected to the internet, while the mobile application must be able to connect to the network in order to exchange information with the server, such as destination and location of the user retrieved via the GPS of the mobile device.

Hardware requirements for both are later specified in [subsubsection 3.4.2](#)

### 3.1.3 Software Interfaces

Supported browsers for the web application should include Google Chrome, Mozilla Firefox, Opera, Safari, Internet Explorer and Edge, while the mobile application must be available on Android, iOS, Windows Phone and Blackberry OS.

The server side of the application requires:

- **Java EE**, to write the server application that perform the travel computation and the database access.
- **MySQL**, to memorize user information and meetings inside a relational database.

The client side of the application requires the latest version of the platform SDK.

### 3.1.4 Communication Interfaces

The client communicates to the server via HTTPS protocol using TCP.

In addition, the system must be able to use the API of other application in order to retrieves weather and news about road conditions or strike.

## 3.2 Functional Requirements

### 3.2.1 Registration

#### Purpose

The main purpose of the *Registration* use case is to provide the user a service which permits the subscription to the system. The user must fill a registration form with his/her personal information and accept the Terms and Conditions of use. After that a confirmation e-mail is sent to the specified e-mail.

#### Functional Requirements

R.1: The system must not accept an already registered e-mail.

R.2: The user must provide the following information:

- name
- surname
- e-mail
- password

R.3: The system cannot allow the user to proceed in the registration process if he/she does not accept the Terms and Conditions of use.

R.4: The system must send an e-mail to the user after he/she submits the form.

R.5: The system must generate a unique link for the registration e-mail.

R.6: The user must be able to exit the form any time.

#### Use Case

The *Registration* use case is analysed in [Table 1](#)

#### Activity Diagram

The activity diagram of the *Registration* use case is showed in [Figure 2](#)

#### Sequence Diagram

The sequence diagram of the *Registration* use case is showed in [Figure 3](#)

<b>Name</b>	<b>Registration</b>
<b>Actors</b>	Non registered user
<b>Entry conditions</b>	–
<b>Flow of events</b>	<ol style="list-style-type: none"> <li>1. The user asks to the system to register to its service.</li> <li>2. The system shows the appropriate form to fill.</li> <li>3. The user fills the form inserting its own information.</li> <li>4. The user submits the form.</li> <li>5. The system checks if the e-mail is unique.</li> <li>6. The system sends to the specified e-mail address a confirmation e-mail with a unique link.</li> <li>7. The user must open the e-mail and click on the confirmation link.</li> <li>8. The system receives the confirmation, saves the data inside a database and notifies to the user.</li> </ol>
<b>Exit conditions</b>	The user is now registered and he/she can login and start to use the service.
<b>Exceptions</b>	Exceptions can occur when requirements R.1, R.2 and R.3 are violated, in this case the system reloads the registration form and goes back to step 2. Registration process is also aborted when the user decides to complete it.

Table 1: *Registration* use case description

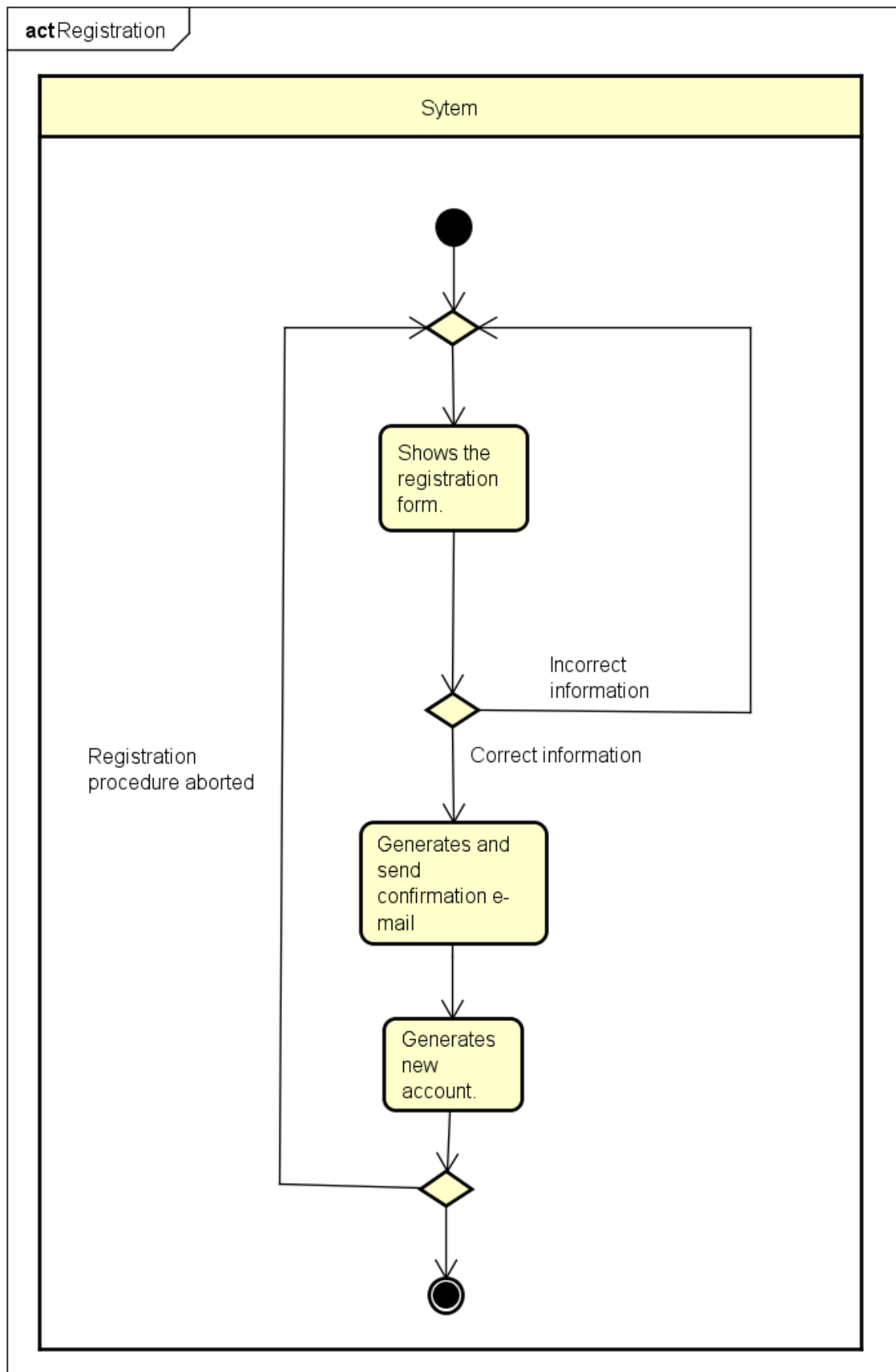
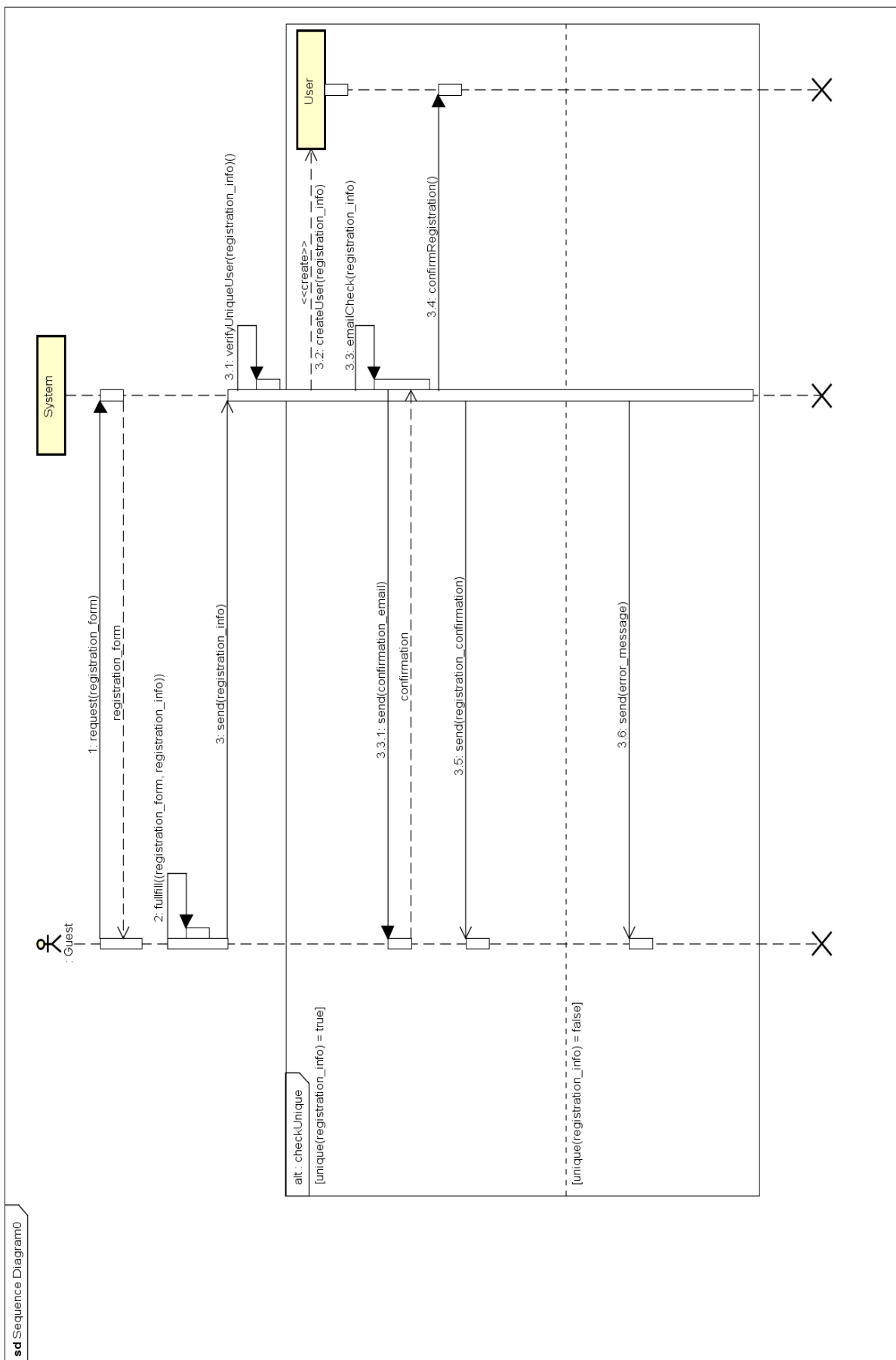


Figure 2: *Registration* activity diagram

Figure 3: *Registration* sequence diagram



### 3.3 Performance Requirements

Without taking into consideration the speed of the internet connection, in order to guarantee an acceptable user experience the following requirements must be satisfied:

- Navigation between pages of the system must happen in 0.5s or less.
- The best travel plan must be computed in 5s or less.
- No limit of registered users in the database.
- No limit of schedulable appointments.
- At least 1000 users must be able to use the service at the same time.

### **3.4 Design Constraints**

#### **3.4.1 Standards Compliance**

The web application must comply with the standards dictated by W3C, while the mobile application must follow the Oracle guidelines for Java programming.

#### **3.4.2 Hardware Limitations**

Minimum system requirements for the two applications:

- Web application
  - 512Mb of RAM.
  - 2Mb/s internet connections.
  - 800X600 screen resolution.
- Mobile application
  - 1Gb of RAM.
  - 3G UMTS internet connections.
  - 100Mb of free space.

The system should also be able to process operations in parallel.

### **3.5 Software System Attributes**

#### **3.5.1 Reliability**

Each trip plan computed given the preferences expressed by the user and the weather forecast and strikes must not differ more than 5% from the optimal travel distance or ETA.

#### **3.5.2 Availability**

The system to be must guarantee an availability of no less than 98%.

#### **3.5.3 Safety and Privacy Constraints**

The user oversees his/her own security while travelling and must grant access to the current location, information about former trips and personal data are stored but only the user itself has access to them.

#### **3.5.4 Maintainability**

The system must be developed in such a way that future implementation of new features and changes to existing ones can be done seamlessly, in other words the system has to be modular and scalable.

#### **3.5.5 Portability**

As already mentioned in [subsubsection 3.1.3](#) the software must be available on different configurations, it must be as environment independent as possible, meaning that it has to work on different platforms with the minimum amount of changes to the software itself.