

Politecnico di Milano

A.Y. 2017/2018

Software Engineering 2: ***Travlendar+***

**Requirements Analysis and Specification Document**

Matteo Biasielli - Emilio Capo - Mattia Di Fatta

v. 1.0

**Table of Contents**

**1. Introduction**

1.1. Document purpose..………………………………………………...………………….3

1.2. Description of the problem………………………………………………………….3

1.3. Actors………………………………………………………………………………………...3

1.4. Goals…………………………………………………………………………………………..4

1.5. Definitions, Acronyms, Abbreviations………………………………………….4

1.5.1 Definitions……………………………………………………………………….4

1.5.2 Acronyms………………………………………………………………………...4

1.5.3 Abbreviations…………………………………………………………………..4

1.6. Reference Documents…………………………………………………………………4

1.7. Document Structure……………………………………………………………………5

1.8. Revision History…………………………………………………………………………5

**2. Overall Description**

2.1. Product Perspective…...……………………………………………………………….6

2.2. User Characteristics……………………………………………………………………6

2.3. Assumptions, Dependencies and Constraints……………………………….6

2.3.1 Assumptions……………………………………………………………………6

2.3.2 Constraints……………………………………………………………………...7

**3. Specific requirements**

3.1. External Interface Requirements

3.1.1. User Interfaces

3.1.2. Hardware Interfaces

3.1.3. Software Interfaces

3.1.4. Communication Interfaces

3.2. Functional Requirements

3.3. Performance Requirements

3.4. Design Constraints

3.4.1. Standard Compliance

3.4.2. Hardware Limitations

3.4.3. Any other constraints

3.5. Software System Attributes

3.5.1. Reliability

3.5.2. Availability

3.5.3. Security

3.5.4. Maintainability

3.5.5. Portability

Formal analysis using Alloy

Effort spent

References

**1. Introduction**

* 1. **Document purpose**

This document focuses on the requirements analysis for the project Travlendar+. The application’s purpose is to support users in handling out one of the most difficult nowadays’ challenges: organization. No previous versions of this application were developed.

This document is meant to be a reference for any person who has an interest in the project. This includes, but is not limited to, development team members, stakeholders and end users.

* 1. **Description of the problem**

The aim of the project is to create an all-in-one system that unites services that are nowadays offered by various different applications(e.g. Calendar, Travel Scheduler). In order to use Travlendar+, final users should be registered and logged in.

Users should be able to schedule their activities directly through the application and, by taking into account travelling times, constraints and preferences expressed by the user, Travlendar+ should:

* Identify the best mobility option;
* Support the user in buying public transport tickets, if necessary;
* Locate the nearest car or bike sharing, if they represent the best solution;
* Warn the user when a place can’t be reached in the available time.

In general, Travlendar+ should make it easier to organize complex schedules, by finding the best compromises between time optimization and the users’ needs and preferences.

* 1. **Actors**
* **Guest**: a person that is not yet registered or logged in. This actor can only see the main page of Travlendar+, where he’s asked to log in or start a registration procedure.
* **User**: a person that is registered and logged in. This type of actor can access to all the features offered by Travlendar+.
* **Third part systems**: external systems that, through the use of APIs, Travlendar+ can use to provide the services it intends to offer.

*(e.g. Google Maps’ APIs can be used to calculate the time needed to move from a place to another by car, bike, public transport system and on foot).*

*(e.g. a car sharing system’s APIs can be used to locate the available cars and propose to the user to use a car sharing service)*

* 1. **Goals**

Travlendar+ features should match with the following goals:

* [G1] Users should be able to use the system properly.
* [G2] Allow users to schedule an activity.
* [G3] Allow users to set constraints and preferences.
* [G4] Simplify procedures and reduce the number of operations that a user has to do to schedule an new activity.
* [G5] Make sure users are on time at their scheduled appointments.
  1. **Definitions, Acronyms, Abbreviations**
     1. **Definitions**
* **User**: actor that is using the application and may want to access all functionalities.
* **Application**: with the term application we are talking about both the desktop version and mobile version of the Travlendar+ system.
* **Scheduling**: action performed by a user that is adding a new activity to his personal calendar.
* **Activity**: whatever the final user wants to schedule, adding it to the calendar and providing the app with its information.
  + 1. **Acronyms**
* **RASD:** Requirements Analysis and Specification Document
* **UI:** User Interface
* **API:** [Application programming interface](https://en.wikipedia.org/wiki/Application_programming_interface)
* **UXD**: User Experience Diagram
* **UML**: Unified Modeling Language
* **GPS**: Global Positioning System
  + 1. **Abbreviations**
* **[Gn]:** the n-th goal
* **[Rn]:** the n-th requirement
* **[NFRn]:** the n-th non-functional requirement
* **[An]:** the n-th assumption
* **[Cn]:** the n-th constraint
* **[UCn]:** n-th use case
  1. **Reference Documents**
* Mandatory project assignments for the A.Y. 2017/2018 available on the beep’s page of the Software Engineering 2 course.
* Projects examples and other documents available on the beep’s page of the Software Engineering 2 course.
  1. **Document Structure**
* **Introduction:** This is the very first part of the document.

In this section it’s possible to retrieve general information about the project and its goals and about the system that is going to be described with more details in the next sections. For the sake of simplicity and to avoid any possible ambiguity, Acronyms, Definitions and Abbreviations that will be used in the whole document have been specified here.

* **Overall Description:** This section will contain a more detailed explanation of the product perspective, that can be useful to visualize and contextualize the project we’re working on. It also highlights the several assumptions and constraints related to the project. Further information about the application’s functions, the user and the requirements will be clarified.
* **Specific Requirements:** This section contains more details and examples about the Graphic User Interface and specifies which are the Communication Interfaces that must be supported by the end-user devices in order to make the application work properly.

In addition, functional requirements will be defined with more details and they will be mapped into goals.

Several UML diagrams will be displayed in this section.

* **Formal Analysis using Alloy:** The source code of an Alloy 4.2 specification of the system.
* **Effort Spent:** Information about the number of hours each group member has spent working on the RASD document.
  1. **Revision History**
* **v. 0.1 [05 Oct 2017]**: added the whole “Introduction” section.
* **v. 0.2 [07 Oct 2017]**: added part of the “Overall Description” section.
* **v. 0.3 [08 Oct 2017]**: completed the “Overall Description” section.
* **v. 1.0 [09 Oct 2017]:** general group revision of the first two sections.

**2.0 Overall Description**

This section includes a summary of the major functions provided by the system, the user characteristics, the constraints and the assumptions over the domain.

**2.1 Product Perspective**

Since the application can be used both on desktop computers and mobile devices and since a user may want to access his calendar from both kind of devices at the same time, data cannot be stored locally. The system we’re going to develop will then consist of an application for the end user and an application for the central server. On the server, users’ data, preferences and schedules will be stored in a Database.

Furthermore, our product needs to be perfectly integrated with some pre-existing systems, such as other car sharing and bike sharing systems, through their exposed APIs.

**2.2 User Characteristics**

As specified in the “Actors” section above, there is no relevant distinction that we have to make between our expected end users, simply because there are no particular kinds of users we are expecting use the application, once it is ready.

Users don’t need any specific knowledge to make a good use of Travlendar+, and, moreover, users’ main life occupations are not really relevant because our application can be used to schedule every kind of appointment, from business meetings to a football match with friends.

* 1. **Assumptions, Dependencies and Constraints**
     1. **Assumptions**
* **[A1] Ubiquity:** Users cannot schedule two different activities at the same time of the same day. This means that the second activity must be scheduled on a different time of the day, or the first one must be deleted.
* **[A2] GPS precision:** to suggest the best mobility option, Travlendar+ has to acquire the position of the user though GPS, if the user hasn’t inserted a starting point manually. We assume that those data have a maximum error of 10 meters.
* **[A3] Email:** Users’ declared emails are supposed to be currently in use. When a user registers, his email is verified. From that moment, then, we assume that the user will not stop using and regularly checking that mailbox.
* **[A6] Accidents:** Accidents and delays are frequent but most of the times online systems that calculate shortest mobility options get to know about them rapidly and take them into account for the solutions they propose. We assume that, if a user is following one of the mobility options given from Travlendar+ (see [A7]), unpredictable accidents that may cause delays have a rate of 10%. This means that the rest of the times (90%) users will be on time at their appointments.
* **[A7] Mobility options:** Users will follow the mobility options proposed by Travlendar+.
* **[A8] Refund policy**: The purchase of a ticket cannot be withdrawn.
* **[A9] Internet**: Users have access to the Internet.
* **[A10] Third part reliability**: The Third part systems we rely on will provide us the service we need at least 99,9% of the time.
  + 1. **Constraints**
* **[C1]** Confidential data inserted by the users must be stored a secure way, according to the actual privacy laws.
* **[C2]** Users’ GPS position can’t be acquired if the user himself didn’t give his consent.
* **[C3]** Failures can happen, but when they happen the system must be able to restart from its status before the failure.

**3.0 Specific Requirements**

**3.1 External Interface Requirements**

**3.1.1 User Interface**

**3.1.2 Hardware Interface**

# 3.1.3 Software Interface

The system will have to rely on a third part map and geo localization service in order to provide users with more precise and customized notifications and directions. To furnish Travlendar+ users with a high-quality service Google Maps (<https://developers.google.com/maps/>) and Google Geolocation (link to: [Google Geolocation API](https://developers.google.com/maps/documentation/geolocation/intro)) APIs will be used.

Additionally, in order to store users’ login credentials and calendar’s data, the application will make use of a commercial DBMS (Database Management System). For this purpose MySQL (Version: 5.7.19, available at <https://www.mysql.com)> has been chosen.

Moreover, to provide some functionalities Travlendar+ has to interface with an external web server.

Furthermore, the system needs to interface with some third part apps and services:

* a weather forecast service in order to provide more advanced and accurate suggestions and instructions. Yahoo! Weather APIs will be used (<https://developer.yahoo.com/weather/> ).
* Third part bike sharing apps and websites such as Ofo and Mobike, for the Milan Metropolitan Area, to be integrated in the application.
* Car sharing service Car2Go APIs to integrate this kind of function in Travlendar+.
* Third part local public transportation apps and website for ticket purchasing.
* Uber APIs in order to integrate this kind of service in Travlendar+ [assumption of agreement with companies]

Furthermore, The Travlendar+ application has been thought and will be developed in order to work on almost any mobile device (supported mobile OS: iOS, Android) and with all best-known web browsers (supported web browsers: Safari, Google Chrome, Internet Edge, Safari).

**3.1.4 Communication Interface**

The entire system will make use of TCP-IP protocol for Internet communication with the web server, the DBMS and users.

**[UC1]** Login

|  |  |
| --- | --- |
| *Name* | Login |
| *Actors* | Guest User |
| *Entry Condition* | The user is already registered and just needs to log in the application. |
| *Flow of events* | * The user launches either the mobile or desktop version of the application * The application homepage is displayed (for further information see user interface samples) * The user must provide his username and password and click on the proper button * The application queries the DB to validate credentials and gives a response to the user |
| *Exit condition* | The user successfully gains the access to the application. |
| *Exceptions* | If the user forgets his password, he can start the password recovery routine.  Moreover, if either the password provided is not correct or the username does not exist, the system notifies the mistake with an error screen. |

**[UC2]** Sign up

|  |  |
| --- | --- |
| *Name* | Sign up |
| *Actors* | Guest User |
| *Entry Condition* | The user is not yet registered, so he should proceed with the registration routine in order to use Travlendar+ services. |
| *Flow of events* | * The user launches either the mobile or desktop version of the application * The application homepage is displayed (for further information see user interface samples) * The user must choose the sign-up option, clicking on the proper button * The system shows the sign-up form asking for the mandatory information that is: username, password, email or Facebook/Google account * Data are sent and stored in the DB |
| *Exit condition* | The user successfully completes the registration routine and gains the access to the application |
| *Exceptions* | If the user decides to interrupt the routine, the application homepage is showed again.  Moreover, if any mandatory information to complete the registration is incorrect or missing, an error screen is displayed. |

**[UC3]** Password Recovery

|  |  |
| --- | --- |
| *Name* | Password Recovery |
| *Actors* | Guest User |
| *Entry Condition* | The user is registered and needs to use the password recovery routine. |
| *Flow of events* | * The user launches either the mobile or desktop version of the application * The application homepage is displayed (for further information see user interface samples) * The user clicks the password recovery button * The system starts the password recovery routine |
| *Exit condition* | An e-mail message is sent at the e-mail address provided during the registration with an attached link to recover the password. |
| *Exceptions* | The user rolls back to the application homepage, interrupting the password recovery routine. |

**[UC4]** New activity

|  |  |
| --- | --- |
| *Name* | New activity |
| *Actors* | Logged-In User |
| *Entry Condition* | The user is logged in and the application is launched. |
| *Flow of events* | * The user clicks on the proper button (for further information see user interface samples) * The application starts the routine and, according to user’s preferences, it asks for activity date, activity location, activity label, notifications’ options, guests and notes about the activity * The system shows the activity added in the calendar and asks for user’s confirmation * Data are sent and stored in the DB |
| *Exit condition* | A message is displayed to notify the user that he successfully added a new activity to his calendar. |
| *Exceptions* | Ether the activity cannot be added due to date, clock (consistency with other activities) issues or the user interrupts the routine. |

**[UC5]** Edit activity

|  |  |
| --- | --- |
| *Name* | Edit activity |
| *Actors* | Logged-In User |
| *Entry Condition* | The user is logged in and the application is launched. |
| *Flow of events* | * The user selects an activity between those in his calendar * He clicks the proper ‘edit’ button (for further information see user interface samples) * The system starts the routine allowing the user to edit different data fields of the activity * The user clicks the ‘save’ button as he finishes to edit the activity * The system saves changes in its server /DB |
| *Exit condition* | A notification makes the user aware that changes performed have been saved. |
| *Exceptions* | Changes performed makes the activity inconsistent with other activities (so user cannot save his changes) or the user himself interrupts the edit procedure. |

**[UC6]** Edit preferences

|  |  |
| --- | --- |
| *Name* | Edit preferences |
| *Actors* | Logged-In User |
| *Entry Condition* | The user is logged in and the application is launched. |
| *Flow of events* | * The user clicks on the proper button (for further information see user interface samples) * User’s profile and his preferences are displayed * User can perform changes on preferences about favorite travel means and distances, third part services to be used and activity scheduling (insert/delete lunch break) * If changes are performed, the system saves them in its server |
| *Exit condition* | The system notifies the user the end of the procedure and that changes have been saved. |
| *Exceptions* | The user exits the edit procedure without save changes. |

**5. Effort Spent**

This section will provide detailed information about the nomver of hours spent on this document.

**Matteo Biasielli**, matr. 893590

|  |  |
| --- | --- |
| **Section(s)** | **Number of hours** |
| 5-oct-17 Introduction | 3 |
| 7-oct-17 Overall Description | 1 |
| 8-oct-17 Overall Description | 1 |
| 9-oct-17 Group work | 3 |
|  |  |
|  |  |
|  |  |

**Mattia Di Fatta,** matr. 893608

|  |  |
| --- | --- |
| 6-oct-17 Introduction | 2 |
| 7-oct-17 Introduction | 2.5 |
| 8-oct-17 Specific requirements | 1 |
| 9-oct-17 Group work | 3 |
| 10-oct-17 Interfaces and start UC | 2.5 |
| 11-oct-17 Add new use cases | 2.5 |
| 12-oct-17 Complete Use Cases tables | 2 |
| 13-oct-17 Class Diagram (to be ended) | 1.5 |

**Emilio Capo,** matr. 899842

|  |  |
| --- | --- |
| 6-oct-17 Introduction | 3 |
| 8-oct-17 Overall Description | 1 |
| 9-oct-17 Group work | 3 |
|  |  |
|  |  |