TRAVLENDAR+

**RASD**

*Requirements Analysis and Specification Document*

Version 1.0

*Greco Sonia*

*Guzzo Francesco*

Release Date: xx.10.2017

***TABLE OF CONTENTS***

[1. INTRODUCTION 3](#_Toc495054605)

[1.1 PURPOSE 3](#_Toc495054606)

[1.2 SCOPE 3](#_Toc495054607)

[1.3 DEFINITIONS 4](#_Toc495054608)

[1.4 REVISION HISTORY 4](#_Toc495054609)

[1.5 REFERENCE DOCUMENT 4](#_Toc495054610)

[1.6 DOCUMENT STRUCTURE 4](#_Toc495054611)

[2. OVERALL DESCRIPTION 4](#_Toc495054612)

[2.1 PRODUCT PERSPECIVE 4](#_Toc495054613)

[2.2 PRODUCT FUNCTIONS 4](#_Toc495054614)

[2.3 USER CHARACTERISTICS 5](#_Toc495054615)

[2.4 ASSUMPTIONS, DEPENDENCIES AND CONSTRAINTS 5](#_Toc495054616)

[3. SPECIFIC REQUIREMENTS 5](#_Toc495054617)

[3.1 EXTERNAL INTERFACE REQUIREMENTS 5](#_Toc495054618)

[3.1.1 User interfaces 6](#_Toc495054619)

[3.1.2 Hardware interfaces 6](#_Toc495054620)

[3.1.3 Software interfaces 6](#_Toc495054621)

[3.1.4 Communication interfaces 6](#_Toc495054622)

[3.2 FNCTIONAL REQUIREMENTS 6](#_Toc495054623)

[3.3 PERFORMANCE REQUIREMENTS 6](#_Toc495054624)

[3.4 DESIGN CONSTRAINTS 6](#_Toc495054625)

[3.4.1 Standards compliance 6](#_Toc495054626)

[3.4.2 Hardware limitations 6](#_Toc495054627)

[3.4.3 Other constraints 6](#_Toc495054628)

[3.5 SOFTWARE SYSTEMS ATTRIBUTES 6](#_Toc495054629)

[3.5.1 Reliability 6](#_Toc495054630)

[3.5.2 Availability 6](#_Toc495054631)

[3.5.3 Security 6](#_Toc495054632)

[3.5.4 Maintainability 6](#_Toc495054633)

[3.5.5 Portability 6](#_Toc495054634)

[4. FORMAL ANALYSIS USING ALLOY 6](#_Toc495054635)

[5. EFFORT SPENT 7](#_Toc495054636)

[6. REFERENCES 8](#_Toc495054637)

# **1. INTRODUCTION**

## *1.1 PURPOSE*

Travlendar+ is a project which aims to create a calendar interface that automatically computes and handles the appointments of a person in order to make sure that the user is never late for his/her meetings and everyday activities.

Travlendar+ also ensures that the user utilizes the best mobility option available at current time, including walking, taking into account possible disabilities of the user, strikes and weather conditions.

The user will be able to give personal constraints (e.g. It won’t be suggested to use the car if the user doesn’t have a driving license; walking distances should be less than a given threshold.. ) by activating or deactivating each travel means.

The application will:

- compute times between meetings and prevent them to overlap.

- keep a spot during the day if the user wants to have lunch in a given slot of time.

- give the possibility to buy tickets of the chosen mobility mean.

## *1.2 SCOPE*

Travlendar+ gives a useful support to users’ life. Achieving the features explained above, the application helps the user making the best decision every day. It doesn’t just keep track of all the appointments and meetings of a person, but it also makes sure that you can actually reach the different locations in time, it tells the optimal way to reach them according to various aspects (weather, personal disabilities, owned mobility means, time of the day..). When the user creates meetings that are unreachable in the allotted time, a warning is created.

It also allows the user to buy tickets for the chosen mobility mean or localize the nearest bike of a specified sharing system.

## *1.3 DEFINITIONS*

## *(1.4 REVISION HISTORY)*

## *1.5 REFERENCE DOCUMENT*

## *1.6 DOCUMENT STRUCTURE*

# **2. OVERALL DESCRIPTION**

## *2.1 PRODUCT PERSPECIVE*

*here we include further details on the shared phenomena and a domain model (class diagrams and statecharts)*

## *2.2 PRODUCT FUNCTIONS*

The most important requirements of the project are:

- the computed way from the starting point to the goal must be optimal:

• it should be the shortest

• it should choose the best mobility mean, taking into account the weather forecast

• it should consider also other World related phenomena.

- the optimal way suggested, should also consider user’s personal preferences, such as minimizing the carbon footprint, etc.

## *2.3 USER CHARACTERISTICS*

*here we include anything that is relevant to clarify their needs*

## *2.4 ASSUMPTIONS, DEPENDENCIES AND CONSTRAINTS*

We suppose that these *domain properties* hold in the considered World:

- the GPS of user’s device is always on and always gives the correct position.

- the application should allow the user to register with his/her personal data, preferences and owned mobility means.

- if there’s a meeting in a given day, there must be at least an optimal way to reach the goal.

- the application refers to Google API in order to faithfully represent delays, arrivals and departures of every public transport mean, possible works on the roads or traffic jams at current time.

- the application should take into account the weather forecast when suggest the optimal way to reach the goal with a given mobility mean.

- during the day, there is a fixed slot of time kept for breakfast/lunch time, chosen by the user.

We assume that these *domain assumptions* hold in the considered World:

- there is one and only one calendar corresponding to one user.

- the user entered his/her personal data in the profile area.

We suppose that these *constraints* hold in the considered world:

- Regulatory policies: the system must require to user the permission to get his position and he has to manage sensible data (position, credentials, e-mail) respecting the privacy law. Furthermore, the system must not use notifications to send SPAM respecting the privacy law.

- It is not possible to create meetings covering the all slot of time reserved for lunch break.

- Hardware limitations:

- USER:

\* 3G connection

\* GPS

\* space for app package

- Interfaces to other applications

- interface with Google API to get information about transport means, traffic jams, mobility schedules, etc.

- interface with Weather Forecast

- the user shouldn’t create overlapping meetings or unreachable ones. A meeting can be unreachable if it not possible to go there in a given slot of time (e.g. it is too far or there are no available transport means).

# **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*

*Definition of use case diagrams, use cases and associated sequence/activity diagrams, and mapping on requirements*

## *3.3 PERFORMANCE REQUIREMENTS*

## *3.4 DESIGN CONSTRAINTS*

### *3.4.1 Standards compliance*

### *3.4.2 Hardware limitations*

### *3.4.3 Other constraints*

## *3.5 SOFTWARE SYSTEMS ATTRIBUTES*

### *3.5.1 Reliability*

### *3.5.2 Availability*

### *3.5.3 Security*

### *3.5.4 Maintainability*

### *3.5.5 Portability*

# **4. FORMAL ANALYSIS USING ALLOY**

*in this section you will include your Alloy model. We require*

*you to comment on the model by discussing the purpose of the model, what you can prove*

*with it and why what you prove is important given the problem at hand. You are also*

*required to show one or more worlds obtained by running your model.*

# **5. EFFORT SPENT**

# **6. REFERENCES**