Software Engineeering 2

Requirement Analysis and Specification Document

Tobias U. Rasmussen – 894983

October 2017

POLITECNICO
MILANO 1863

Contents

1 Introduction **1**
- 1.1 Purpose . . . 1
- 1.2 Scope and Problem Description . . . 1
- 1.3 Definitions and Abbreviations . . . 2
- 1.4 Document Structure . . . 3

2 Specific Requirements **4**
- 2.1 External Interface Requirements . . . 4
  - 2.1.1 Hardware . . . 4
  - 2.1.2 Software . . . 4
  - 2.1.3 User Interface . . . 4
- 2.2 Functional Requirements . . . 4
  - 2.2.1 Travlendar$^+$ User(Logged off) / New User . . . 5
  - 2.2.2 Travlendar$^+$ User(logged in) . . . 5
  - 2.2.3 Developer(Third party developer) . . . 6
- 2.3 Non-Functional Requirements . . . 6
  - 2.3.1 Security . . . 6
- 2.4 Performance Requirements . . . 6
- 2.5 Design Constraints . . . 6

3 Scenarios **8**
- 3.1 Scenario 1 - Making an account . . . 8
- 3.2 Scenario 2 - Making an appointment . . . 8
- 3.3 Scenario 3 - Making several appointments . . . 9
- 3.4 Scenario 4 - Making an unreachable appointment . . . 9
- 3.5 Scenario 5 - Deleting an appointment . . . 9

4 UML **10**
- 4.1 Use Cases . . . 10
  - 4.1.1 Register as a User . . . 10
  - 4.1.2 Log-in . . . 11
  - 4.1.3 Making an appointment . . . 12
  - 4.1.4 Delete an appointment . . . 13
  - 4.1.5 Edit an appointment . . . 14

5 Alloy **15**
- 5.1 Alloy-model . . . 15
- 5.2 Summary . . . 15

6 Summary **16**

7 Work Log **17**

8 GUI-Mockups **18**

1 Introduction

1.1 Purpose

Our team will project a suggested realization of the **Travlendar**$^+$. More specifically the **Travlendar**$^+$ will be an combined web and phone-application. The application are going to work as a combined calendar and travel-companion. If a user makes an appointment in the calendar, they will also be able to specify the exact position of said appointment. The system will then calculate the optimal route to the desired meeting point to ensure that the user get to the appointment in due time. The system also features a significant amount of trip-customization, such as: transport type, carbon-footprint and travel constraints(maybe a max walk distance 1km).
This specific document should be considered as a description of the high-level functionality of the system.

1.2 Scope and Problem Description

In today's society many appointments are at various locations across the city or region where we live. This projects goal is to create a combined calendar and travel-companion. The application must be able to:

I Automatically compute travel time between appointments and notify the user in such a way that the he/she arrives in due time.

II Support the user in their travel, by combining ways of transport and assessing live traffic-updates.

Specific problems:

1. Any registered user may set up a appointment, this can include: time and date of appointment, place, notes, thresh-hold for notification(i.e 5 min before), travel constraints and attendants. For the user to get a suggested travel-route, they have to specify time, date and place.
2. A user may invite other users to any given appointment if and only if they are on each other contact-lists. When invited, the user must acknowledge the invite and verify travel constraints to participate.
3. When an appointment is made, the system guaranties a optimal travelling-route in respect of the users constraints. The system will verify if the appointment and intermediate appointments is reachable from the current location with respect to time.
4. If one or several appointments are unreachable, the user will be notified by a warning.

5. If a location is unreachable with current user constraints, the application will ask the user if it should give an suggested traveling route ignoring the current user-constraints. If granted access the system will ignore all constraints, except private transportation.
6. In addition to basic functionality, the system will be designed such that additional functionality and features may be added after release.

Domain assumptions Some assumptions made to interpret the system and its interaction with the real world:

- The user updates possible means of travel each day. i.e. the user removes car from possible transport if they leave it at home that day.
- Users only choose means of transportation that are available for them.
- Only registered users can call in other users to an appointment and attend other peoples meetings.
- When a user picks a location for an appointment, it is correct.
- Each user has a device with internet connection and GPS at all times
- There will be little or no spam in form of fake/commercial appointment-invites.
- Every user either has or has no internet(4g/3g).

Goals To achieve that the **Travlendar**$^+$ works as intended, we have set some goals that the application/system must satisfy:

1. To let the users easily set up appointments
2. Let users invite other users(friends/associates) to their appointments
3. The application must generate a optimal travelling-route and transport to the meeting point including the time of travel
4. Notify the user in due time before a appointment, such that the user will get there in time.
5. Provide warning if the location of an appointment is unreachable with or without the current user-constraints.

1.3 Definitions and Abbreviations

Definitions:

Software or System - The collection/part of programs which this document describe and together constitute the application itself.

Appointment - A scheduled meeting with one or more persons.

User - Any person that uses the application.

Actor - Any user or program interacting with the application.

Unreachable - The user cant reach the appointment in time.

User-Constraints - Constraints in terms of transport given by the user.

Abbreviations:

API - Application Programming Interface, a set of subroutines, protocols, and tools for building application software.

ETA - Estimated Time to Arrival.

PW - Password,

App - Software application for any mobile device or web-browser.

GPS - Global Positioning System, a world wide service to provide accurate position data.

WIFI - Wireless local area network, here implicit used as internet connection.

1.4 Document Structure

2 Specific Requirements

In this section we will consider the scope and problem description from 1.2. And use these to put down the specific requirements for our software.

2.1 External Interface Requirements

2.1.1 Hardware

Travlendar$^+$ must be granted access to use the device location services, GPS or 4g/3g. Preferably both to provide a more precise estimation of both time needed and the route. In addition the application needs access to internet, either WIFI or 4g/3g. It must also be allowed to raise push notifications on the device which it is used.

2.1.2 Software

The **Travlendar$^+$** is going to use Google maps API to provide both map visualization, ETA's and travelling-routes. This takes care of the map visualisation bit, the calendar are going to be provided by Google-Calendar API. This way the system is composed of 3 different API's, 2 of which are coming from the same producer, which may be an advantage in terms of price, development and support.

2.1.3 User Interface

For a user, the application is going to exist as a single version application. This will be available in 3 different ways:

- **Web-Application** - for everyone that has a web-browser
- **Google-play** - For android devices
- **App-Store** - For iPhone devices

This will make it easy for developers to release patches, and for the user to update the application when a new patch arrives. It is also a very easy way of distributing the software to a larger audience.

2.2 Functional Requirements

The more specific functional requirements regarding the actors of the software are uncovered here.

2.2.1 Travlendar$^+$ User(Logged off) / New User

This actor is not a registered user. This means that they either is registered or not. Either way they must be able to do the following:

1. Register as a new user, if he is not already registered
    - They are not an registered user, and must therefore set up an account to be able to log in.
2. Log in using email and password
    - They have already registered, and need only to log-in to use the app.
3. Perform password recovery
    - They may have forgot their PW, this will send an temporary PW such that they can log-in and reset their PW.

2.2.2 Travlendar$^+$ User(logged in)

This actor is a registered user and are using the app frequently. This actor must be able to interact with the app in such a way that it satisfies goal: 1, 2, 3, 4 and 5(See goal: 1, 2, 3, 4 and 5). Which means that this actor must be able to:

1. Set up an appointment.
    - The application must let the user choose place and time.
    - The system will provide a suggested optimal route by using the users current position.
    - Visualize the suggested traveling route and means of transport, by using the Maps-API.
    - The application will allow the user to invite other users, and notify these of the invite with push-notification.
    - Calculate if the new appointment is reachable from the location given by the last appointment and visa versa.
    - Show a warning window if the appointment is unreachable.
2. Cancel an appointment
    - If the appointment have several attendants, their appointment will not be deleted
3. Update the personal settings
    - Change means of available transport
    - Update personal data

2.2.3 Developer(Third party developer)

Should be able to get a hold of the Travlendar$^+$-API, such that further development may proceed. Also basic access, the same as a Travlendar$^+$ user.

2.3 Non-Functional Requirements

With all the functional requirements listed above, here we will identify the non-functional requirements:

2.3.1 Security

- **Appointment-Notes** - When an appointment is made, the information it contains can only be accessed by the person which made it.
- **Appointment-Information** - When one or several users have an appointment, the appointment information will only be access-able by the attending users.
- **User blocking** - A user has the possibility to block other users from sending them requests, this to prevent spam.
- **Appointment-cancellation** - If a owner of an appointment cancel the appointment, every attendant is notified, however they may choose to delete the appointment or not.
- **Disconnection** - If the application looses internet connection or GPS signal. The user is notified and the system will not provide a suggested route. However it will give the last calculated ETA if it has any, if not it will function as a normal calendar.

2.4 Performance Requirements

The system must be able to take a multiple of request from several different users at any given time. It must also provide an suggested route and ETA in a acceptable amount of time. Acceptable being 1 minute or less. Given the assumption 1.2, this will be taken care of by the Maps-API.

2.5 Design Constraints

Some constraints has to be identified to avoid some common pit-falls:

- **Size** - The size of the application available to devices should be kept to a minimum. So a maximum size of: 15 - 20 MB, this should suffice to give

the developers room for development, and also keep the application light enough

- Using Google Cloudplatform as a database provider, this is an SDK- which makes it easier to customize it to our API.
- The application must be compatible with the most common web browsers.

3 Scenarios

In this section different scenarios will be identified. This to create a clear picture of the use-cases, and to clearer see the actors interacting with the system. Here we introduce Roy:
Roy is a working man, more precisely an building-consultant. Roy does a lot of travelling throughout the city during to visit clients the day. Since Roy mostly relies on public transport, he must always use some part of the day planning the travel between each client. Roy stumbles upon the Travlendar$^+$ on Google-play during his lunch on a Monday. Roy decides to try this out, since it seem to fit his situation perfectly

3.1 Scenario 1 - Making an account

Roy have now downloaded the Travlendar$^+$. The first thing he encounters is the log-in screen. There are two possibilities: Sign-in and Sign-up. He creates a new account by navigating to sign-in. Here he must post his name/sir-name, email-address and make an password. Roy fills in the needed information, accept the terms and is automatically returned to the log-in screen. Roy now tries to log-in, by typing his email and pressing sign-in.

3.2 Scenario 2 - Making an appointment

Roy is now signed in. He is met with a screen showing a pop-up window in which he can turn different types of transportation on/off. He turn on every type of public transport and clicks save. Now the screen shows a basic calendar, just like the one Google has. This is familiar to him and now he wants to set up his first appointment. He creates a new appointment, and is now asked to fill in some basic information about the appointment: Time, date, position, Name, pre-notification time(10,15,30 min), duration and Notes. Obligatory fields are marked with: $*$. Roy fills in Time, date, appointment-name and a position. He is supposed to meet someone after lunch 20 minutes away. Roy saves the appointment which now is visible in the calendar.
Five minutes later the phone makes a sound and vibrates, Roy picks it up and sees a push notification: (The bus to your appointment leaves in 10 minutes. Click to watch route.). Roy clicks on the push notification and a map appears showing a map with a route plotted in. In the bottom of the screen there are a section showing the start and stop-bus-stations with a details button.

3.3 Scenario 3 - Making several appointments

The next day Roy decides to try out the application a little bit more, he adds 2 appointments. The first appointment is meeting up at work, the next one is the first client 2 hours after work. When he has made the appointments he want to double check the time of the appointments. He checks the second one and see that he can preview the travelling route, he clicks preview and see which bus he has to take, to reach to the metro from his workplace and when he have to leave work.

3.4 Scenario 4 - Making an unreachable appointment

Right after checking his appointment Roy gets a call from his client, the client want to move the meeting 1 hour earlier in the morning. Roy accepts and changes the time of the appointment. When he presses save a pop-up window appear: Warning! This appointment is unreachable with your transport-settings from your last appointment. Roy see that he can press: OK or Cancel. Roy presses cancel and calls his client to reschedule once again.

3.5 Scenario 5 - Deleting an appointment

When asking his client to reschedule, the client see no other option than to take the meeting via phone. Roy accepts and goes into his Travlendar$^+$ to delete his appointment. He click on the appointment highlighted in the calendar and see the option Delete at the bottom. He clicks it, affirms the yes no prompt. And is returned to the calendar.

4 UML

This section contains UML models. Thus including: use cases, State-charts and Sequence-diagrams.

4.1 Use Cases

This section contains all the use-cases for our system both model and actor. The following use-cases are derrived from the scenarios listed in section 3

Actors

- Un-registered User
- Registered User
- 3rd-party developer

4.1.1 Register as a User

Name	**Name**
Register User Actor	Un-registered Travlendar$^+$ User
Pre-conditions	• Actor not yet registered as a user

Flow of events	1. The user is greeted with the log-in page
	2. The actor presses on sign-up button
	3. Actor is shown the sign-in form
	4. Actor fills in information:
	  • Email
	  • Password
	5. Actor presses sign up
	6. Actor is returned to log-in page

Postcondition	• Actor is now a registered Travlendar$^+$ User
Exception	• Actor tries to log in before registering Error message shown • Actor tries to sign up without filling in requisite information Actor remains on the sign-in form untill all prerequisites are good

4.1.2 Log-in

Name	**Name**
Log-In[h] Actor	Registered Travlendar$^+$ User
Pre-conditions	• User already signed up • User is not yet logged in

Flow of events	1. The user is greeted with the log-in page
	2. The user fills in log-in information
	3. System checks log-in information:
	  • Email
	  • Password
	4. The user is greeted with the Travlendar$^+$ homepage

Postcondition	• The user is successfully logged in
Exception	• The user forgot to fill in information An error message is shown • The user filled in wrong information An error message is shown • First time log-in User must fill in transport-settings

4.1.3 Making an appointment

Name	**Name**
Make Appointment Actor	Registered User
Pre-conditions	• User is already logged in
Assumptions	• User will not double-book an appointment • User will not make an appointment with a date in the past

Flow of events	1. The user is greeted with the home-page
	2. The user press on add-button
	3. The user is asked to fill the appointment-form
	  • Name
	  • Choose Time and date(pre-set:Current time +1 hour)
	  • Position
	  • Transport-constraints
	  • Pre-notification time(pre-set:10 min)
	4. User press save
	5. System set alarm based on planned route
	6. User is returned to home-page

Postcondition	• An appointment is now made • An alarm is now set to inform the user in time
Exception	• No appointment-name Error shown • No means off transport chosen • The user leaves position empty Warning raised before user can save • Position unreachable in time Warning raised before user can save • Next appointment(if any) becomes unreachable Warning raised before user can save

4.1.4 Delete an appointment

Name	**Name**
Delete appointment Actor	Registered User
Pre-conditions	• User is already logged in • Already made an appointment
Assumptions	•
Flow of events	1. The user is greeted with the home-page 2. The user press an appointment 3. The user see the appointment-information screen 4. User press delete appointment 5. User must confirm that he want to delete 6. System deletes the appointment 7. User is returned to home-page
Postcondition	• An appointment is now deleted
Exception	• There are no appointments to delete

4.1.5 Edit an appointment

Name	**Name**
Edit appointment Actor	Registered User
Pre-conditions	• User is already logged in • Already made an appointment
Assumptions	•
Flow of events	1. The user is greeted with the home-page 2. The user press an appointment 3. The user see the appointment-information screen 4. User press edit button 5. User is prompted with earlier filled appointment-form 6. User changes information to his will 7. User presses save 8. User is returned to home-page
Postcondition	• An appointment is edited
Exception	• The new position is unreachable Warning raised before user can save • The new time-constraint/transport-constraint makes the appointment unreachable Warning raised before user can save • The new time-constraint/transport-constraint makes succeeding appointment unreachable Warning raised before user can save

5 Alloy

5.1 Alloy-model

5.2 Summary

6 Summary

Hopefully this has been a useful short guide. Please also check out the tex source code. For more inspiration check out [1]. Also, since LaTeXis so extensively used, StackExchange and Google generally has the answers you are looking for.

7 Work Log

Date:	Name:	Hours:	Total:
14.10.2017	Tobias Rasmussen	3	3
15.10.2017	Tobias Rasmussen	5	8
16.10.2017	Tobias Rasmussen	3	11
17.10.2017	Tobias Rasmussen	1	12
21.10.2017	Tobias Rasmussen	5	17
23.10.2017	Tobias Rasmussen	8	25

Table 1: Table showing worklog

8 GUI-Mockups

This section contains GUI-mockupsl. These illustrations are not intended as a final design, only as means of illustrating the functionality and use of the software.

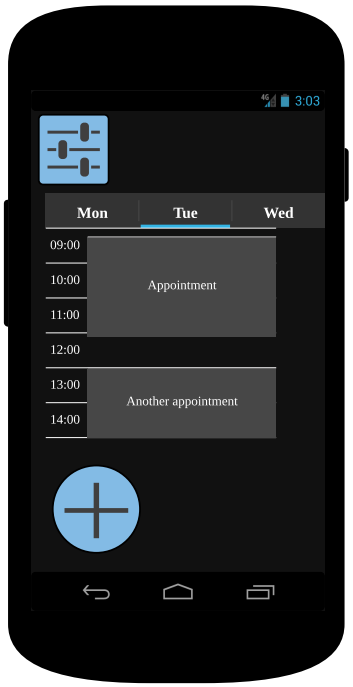

Figure 1: The application home-page

Figure 2: Make a new appointment

Figure 3: Choose type of transport

Figure 4: Appoinment information

Figure 5: Edit an appointment

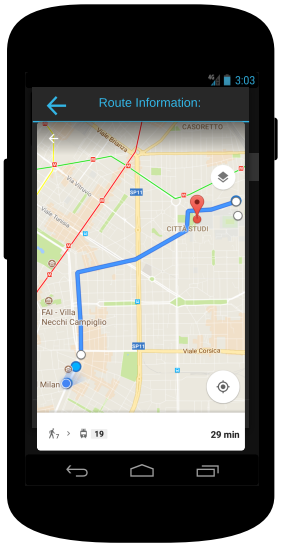

Figure 6: Travel route map

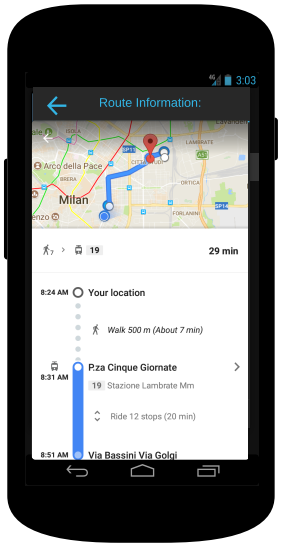

Figure 7: Travel route information

References

[1] Wikibooks latex. https://en.wikibooks.org/wiki/LaTeX. Accessed: 2016-08-30.