Politecnico di Milano
A.A. 2017-2018
Software Engineering II project

Travlendar+

Requirements **A**nalysis
and
Specifications **D**ocument

Mirko Mantovani (893784), Matteo Marziali (893904)

October 21, 2017

Contents

1 Introduction

1.1 Purpose

The main purpose of Travlendar+ is to create a software that allows users to easily manage their daily meetings and commitments, by providing some useful features such as finding the best means of transport to reach the appointment place and easily know the quickest route available to be punctual. Specifically, our goals consist in:

G1 Providing a calendar and the possibility to memorize events and appointments on it.

G2 Automatically computing and accounting for travel time between appointments to make sure that the user is not late for them.

G3 Automatically generating warnings to notify the user that at least two meetings are overlapping (this means the timings are incompatible, in other words the user can't reach in time the second meeting if the first meeting finishes on time).

G4 Providing routes and travels according to user preferences about the preferred/prohibited travel means and daily breaks set.

G5 Possibility to add reminders for a meeting in order to prevent the users forgetting their appointments.

G6 Allowing the users to modify appointment schedules.

G7 Automatically notifying people involved in a specific meeting if the user is late and has selected this feature previously.

On the other hand, the purpose of this paper is to define in a detailed way all the functions and requirements of our application. In doing this, we start focusing on a brief overview to characterize the product with relevance to its interaction with the world, then we will proceed deeply in analysing which functions are relevant and should be provided, and which requirements are needed to the stakeholders.

1.2 Scope

Even if we are very confident about the success of our idea, initially, Travlendar+ will have a restricted domain, indeed we will experiment it only in the Italian city of Milan. In order to provide the most complete assistance, Travlender+ will suggest different paths and a wide range of transports such as bike, even shared, your car or a shared one, taxi, bus, and also.. your feet! It goes without saying that to organize this kind of service in the most valuable way we must interact with a huge number of local institutions which provide different services in town.

1.3 Goals

1.4 Definition and Acronyms

1.4.1 Definitions

- **App:** this is the abbreviation for application, in particular this term is used meaning a mobile application.
- Delay notification function: this phrase refers to the function which allows to notify the participants of a meeting through an email in case the user is late.
- **Travel:** a travel is any suggested path that goes from the starting point to the meeting location.
- **Route:** this term is used as a synonym of travel.
- **Warning:** warning is the word used to define the conflict between two meetings.
- **Calendar:** the calendar contains the list of meetings and is grouped by day.
- **Meeting:** is an important keyword of the application, it includes all the informations of an appointment.
- **Reminder:** a reminder is a sort of an alarm triggered at a certain time before an appointment is starting.

1.4.2 Acronyms

- ETA: estimated time of arrival: the time, estimated by the system, that the closest available taxi will take to get to the starting location of the ride.
- API: application programming interface; it is a set of routines, protocols, and tools for building software applications on top of this one.

1.5 Revision

v1.1 after project class meeting on 11/10/17 Added lists to goal, functions and assumptions; Revised section 1 and 2

1.6 Actors

- *Guest*:
- *Logged in users*:

1.7 References

- The document with the assignment for the project
- The IEEE Standard for SRS

1.8 Document Structure

This document is structured in three parts:

- Introduction: In the first introductory section, we give a short description both of the goals and of the environment which our app has to deal with. Moreover, we explain some notes useful to understand and read the whole paper.
- Overall Description: gives a general description of the application, focusing on the context of the system, going in details about domain assumptions and constraints. The aim of this section is to provide a context to the whole project and show its integration with the real world and showing the possible interactions between the user, the system and the world itself.
- Specific Requirements: this section contains all of the software requirements to a level of detail aimed to be enough to design a system to satisfy said requirements, and testers to test that the system actually satisfies them. It also contains the detailed description of the possible interactions between the system and the world with a simulation and preview of the expected response of the system with given stimulation.
  Finally, we express the requirements through the Alloy model, which allows us to define the interactions, the functions and the constraints that characterize Travlendar+ using a formal language. The document ends with a short note about the effort spent in producing it and at last you can also find useful references.

2 Overall description

2.1 Product perspective

Our idea is to create a personal companion application to help users managing and organizing their daily life. According to this intention, we would like to realize an extremely friendly user interface and a lightweight software in order to make Travlendar+ affordable to many people and runnable by many devices. In order to make use of every functionality the devices require GPS service and an internet connections for most of the services. Since Travlendar+ is going to offer many routes depending on different travel means, it will necessarily have to interact with many institutions such as public transport and car/bike sharing providers. This aspect will affect both the software and the hardware design. Indeed, it is necessary to query data about the shared cars, bikes and the taxis location around the city and to retrieve information about trains and buses schedules. Hence, our system must be very fast and dynamic to support a huge number of query in a few seconds, moreover to interview external databases it's strictly required that the users have an active internet connection. Concerning the hardware, we intend to have a database which only contains username and password of all our customers. For sure, it seems useless having a database for those kinds of data but in our idea this choice allows the app to be updated and improved easily in the future, for instance saving on the database clients' routes and meetings.

2.2 Product functionalities

F1 Signup and Login: Travlendar+ users must sign up the first time they intend to create a meeting and further usages of the app will require a login to access all its functionalities.

F2 Meeting creation: This is the most important function of the app, it allows to generate an event related to an appointment. It requires the user to define all the details such as date, time, location, starting point, preferences etc.

F3 Preferences set up: An important feature of Travlendar+ consists in allowing the user to filter out specific routes depending on some constraints about the travel, or to set break-dedicated time slots.

F4 Warnings management: In case the user seems to be late, the app generates a warning to alert him. Hence, he can either ignore it or postpone the appointment.

F5 Delays management: If the app had noticed, according to the estimated travel time, that the user is in late, and he previously had inserted the email address of the meetings participants, Travlendar+ would notify them about the delay.

F6 Route generation: the main hidden function of Travlendar+ is to automatically compute and suggest to the user the best travel among those which fit the preferences he has selected.

F7 Reminder management:

F8 Recurrent events management: The smartest function Travlendar+ will offer; it consists in allowing the user to select events to be rescheduled periodically just creating one meeting. Done this choice, the app. Automatically manages to reschedule the specific meeting according to the period that the user establish, for instance one week, one month

2.3 User characteristics

According to our idea, Travlendar+ does not have a specific customers range, it is supposed to be used by both male and female, whatever their age is. Obviously, considering that we intend to produce a mobile application, people who want to use Travlendar+ should be familiar with a portable device like a smartphone or a tablet. For sure, users should respect several requirements due to the travel means they are going to take. For instance, when cars are selected as active in the means of transport list, the user is supposed to have a valid driver licence, taking the bus is allowed only with an appropriate ticket. Moreover, users interested in dealing with Travlendar+ services must have an e-mail address, primarily due to register and authenticate themselves, secondly to use the delay notification function.

2.4 Assumptions and dependencies

2.4.1 Domain Assumptions

- The user is supposed to attend every meeting
- A meeting should not take more time than expected.
- If a user has a meeting in a specific location, he's supposed to be there at the end of the meeting, and the app will compute a route based on that information
- The Metros and buses are supposed to be on time, travel times are calculated based on the expected route duration.

2.4.2 General Assumptions

A1 Signup and Login

Considering that the assignments provided do not say much generally about users without any reference to a possible signup or login, we assume that the registration is mandatory to create the first meeting, then every access to the app requires the login to manage each event saved. Please note that login parameters could be memorized to save and recover easily a user instance.

A2 Meeting management

According to the requirements, we want to develop a system which allows the user to set his preferences with regards to the travels. Moreover, we decide that he can also cancel or anticipate/postpose an event, assuming a previous reschedule agreement among the participants. It goes without saying that an appointment can also be modified, this means that a user can change either the starting location or the arrival location, the hour, the date and the other details chosen during the creation of the meeting, always making the same rescheduling agreement assumption.

A3 Warnings

Our assumptions about the warning are the following: when the system generates a warning, the app allows the user to modify the related event that could be cancelled or delayed. In case of the user postposes the meeting, if he provided the email addresses of the other people involved in the appointment, Travlendar+ automatically will notify them that a change occurs.

A4 Routes

Concerning the routes, we decided to manage them in this way. The system generates different routes according to the user preferences, it will be the user itself to decide which itinerary fits better with him among the alternatives.

A5 Preferences

As far as the preferences are concerned, we decided that they belong to a user instead of a meeting. This means that a user cannot define different preferences for each meeting, while they are valid for every appointment.

3 Specific requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

3.1.2 Software Interfaces

Considering the domain of our application, we have decided to integrate in our project some software components to create an easier and more powerful product.
In order to provide an excellent navigation service, we thought to adopt the Google Maps API, to retrieve the user location through the Google Maps' servers and databases.
In addition, we have noticed that the most complex computations which Travlendar should effort are those related to calcuating the best route. This idea suggested us to use the API of several travel mean sharing services, such as Mobike or CarToGo, to support this crucial phase. We are sure that these APIs would have allowed us to query the databases and to retrive precise information in the quickest and easiest way.
The same reasoning is applicable to forecast, needed to avoid certain routes in case of particular wheater, indeed wheater information from a specific provider are supposed to be rietrived through its APIs.
Finally, APIs of public transport agencies are required, to retrieve real time information about buses,trains and taxis and to mantain the app. up to date with relevance to all the related news, for instance about strikes.

3.2 Functional Requirements

- Logging in whitout being signed up is prevented. (f1)
- Logging in, being already logged in, is impossible. (f1)
- Signing up, being already signed up, is impossible.(f1)
- Creating a meeting with the same schedule of an existent one is not allowed. (f2)
- Scheduling a meeting during break pauses, setted in preferences, is prevented. (f2)
- it is impossible to generate an appointment in the past or in an invalid date. (f2)
- Meeting creation requires name,date,hour,location and a starting point to be defined properly.(f2)
- At least one mean of travel must be selected.(f3)
- Every lunch break must last at least 30 minutes.(f3)

- If rain or snow are in the forecast, travel by bus is preferred. (f6)
- In case of strikes, routes involving the related travel mean are not considered. (f6)
- Reminders must be setted before the scheduled time of the related events(f7)

3.3 Non functional requirements

- Simple User Interface: The user interface has to be as simple and intuitive as possible, the application should allow an average user to set up an account and start using the application understanding its functionality in no more than a dozen minutes.
- Portability. The client has to be compatible to all the major hardware and software platform on the market, this is accomplished using the web application solution presented early.
- Performance. The application should be able to calculate shortest paths very quickly in order to let the user choose the one that better fits his needs right after setting up the meeting.
- Reliability. The system should be able to guarantee the service independently of the time, 24/24, 7/7. Thus, the used services should be always available, however, since this is not a critical application, brief unavailability could be acceptable.
- Data integrity, consistency and availability. System data have to be always accessible. Hence the system should always provide a reliable access to them in normal condition. They also have to be duplicate in order to avoid data losses in case of system fault.
- Security. Hashed password should be stored in the database in order to guarantee a high level of privacy to the users. Sensible data such as meetings details will probably be stored locally on the user device, a possible encryption might be considered but it's not a priority

3.4 The world and the machine

The first model of the system to be presented is the model "The world and the machine" by M. Jackson and P. Zave. This model highlights the division between phenomena that happen entirely either in the world or in the machine, and those that are shared between the two of them.

immagine

3.5 Scenarios

3.5.1 Scenario 1

3.5.2 Scenario 2

3.5.3 Scenario 3

3.6 Use Cases

This section contains all the use cases initially described with the use cases UML model of the whole system.

3.6.1 User Page use cases

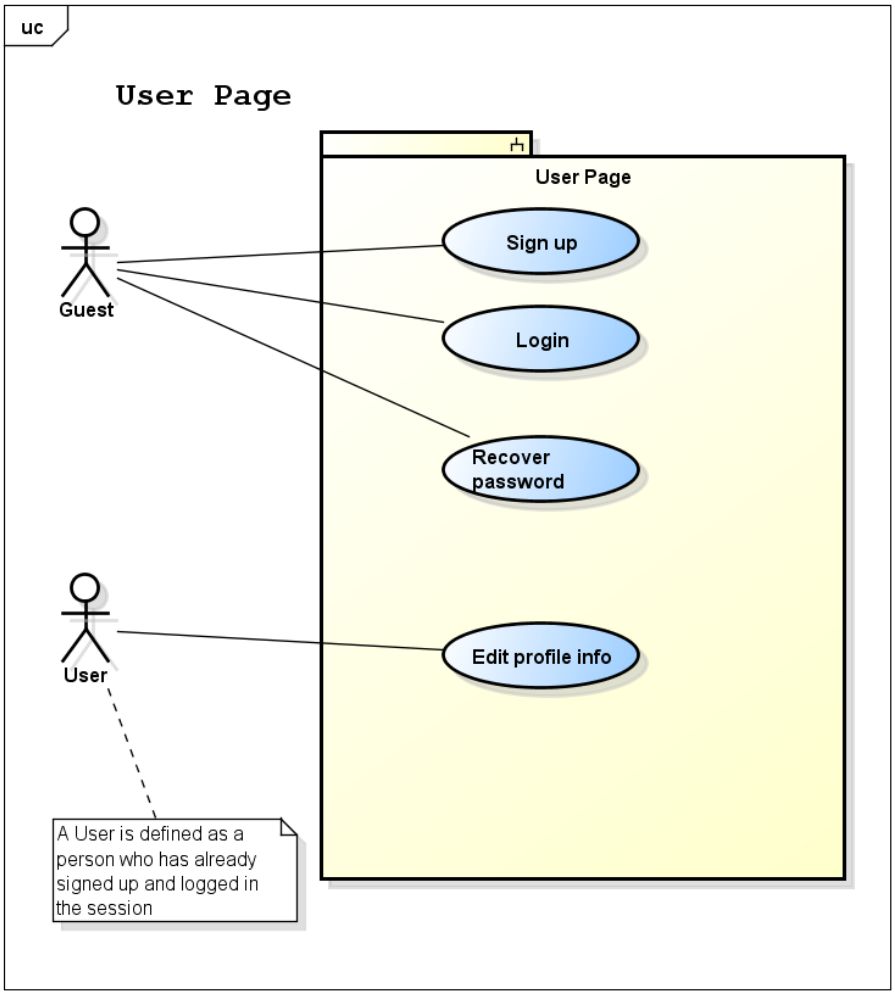

Figure 1: Use cases relative to the user registration and authentication

3.6.2 Sign up

Name	Sign up
Actors	Guest
Entry conditions	None
Flow of events	• The guest reaches the registration page containing the relative form • The guest fills up the form and clicks on "Sign up" to complete the process • The system redirects the user to his profile page and sends a confirmation email.
Exit conditions	The guest has successfully registered in the system.
Exceptions	The guest left an empty field or typed something wrong an error message is displayed and the user is asked to fill the form again.

Table 1: This is my one big table

3.6.3 Login

Name	Login
Actors	User
Entry conditions	The user has already registered.
Flow of events	• The user reaches the login page containing the relative form • The user types the username and password in the login form and click on "Login" button. • The system redirects the user to the application homepage.
Exit conditions	The user has access to the application functionalities.
Exceptions	Username and password didn't correspond or the username didn't exist ,an error message is displayed and the user is asked to fill the login form again.

Table 2: This is my one big table

3.6.4 Password Recovery

Name	Recover Password
Actors	User
Entry conditions	The user has already registered.
Flow of events	• The user reaches the login page containing the relative form • The user clicks on "Password recovery" button and is redirected to the password recovery page. • The user inserts his email and clicks on "reset password". • The system sends an email to the user with a link and instruction to reset the password. • The user chooses and types a new password and confirms. • The system redirects the user to the login page.
Exit conditions	The user has changed his password
Exceptions	The inserted email doesn't match any user in the database, it is displayed an error message and the user is asked to retype a valid email.

Table 3: This is my one big table

3.6.5 Schedule Management use cases

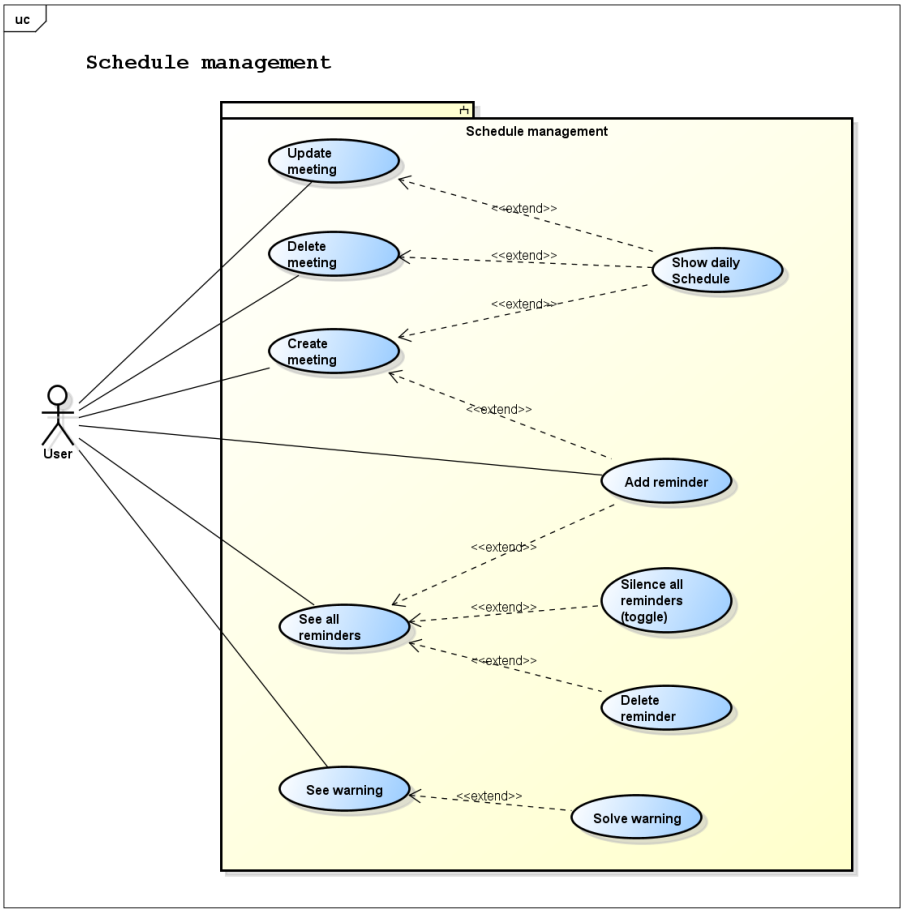

Figure 2: Main use cases showing the functionalities of Travlendar+ application relative to the meetings creation and management

3.6.6 Meeting Creation

Name	Creation of a meeting
Actors	User
Entry conditions	The user is logged in and is in the main page.
Flow of events	• The user clicks on "Create meeting" button and he is redirected to the page with the input form to create a meeting. • The user fills up the form with the meeting information (title, location, description, date and time, flags, etc...). • The system checks whether the meeting at the specified time and date is possible according to the user preferences and the current daily schedule, the optimal route is proposed. • The user is then redirected to the main page.
Exit conditions	The new meeting with the calculated optimal route is added to the user Calendar. In the case of the incompatibility of a meeting with others a warning is created.
Exceptions	The information inserted is wrong or some information is missing: a corresponding error is displayed and the user is asked to modify the inserted information accordingly.

Table 4: This is my one big table

3.6.7 Reminder Addition

Name	Addition of a reminder
Actors	User
Entry conditions	The user is logged in and is on the page of a meeting
Flow of events	• The user clicks on "Add reminder" button and he is redirected to the page with the input form to add a reminder. • The user fills up the form with the type of reminder and the time he wants to be reminded of the upcoming meeting. • The system adds the reminder and the user is redirected to the relative meeting page.
Exit conditions	The reminder is added to the meeting
Exceptions	There exists already an identical reminder and it is not added to the meeting

Table 5: This is my one big table

3.6.8 Warning Solving

Name	Solving a warning
Actors	User
Entry conditions	The user is logged in and is in the page of a warning
Flow of events	• The user clicks on "Solve warning" button and he is redirected to a page that lets him choose how to solve the conflict: the timing of two overlapping meetings can be changed, or one of the two meetings has to be canceled; • The user solves the conflict the way he wants and clicks on the button "Done". • The system checks whether the conflict has been solved and the user is redirected to the warnings page.
Exit conditions	The warning has been solved and is deleted from the system and from the list of warnings in the corresponding page
Exceptions	The warning was not solved after the user's modifications, the unresolved warning will still be present and a message stating that the conflict wasn't successfully solved is displayed. The user is redirected to the warning page.

Table 6: This is my one big table

3.6.9 User Preferences use cases

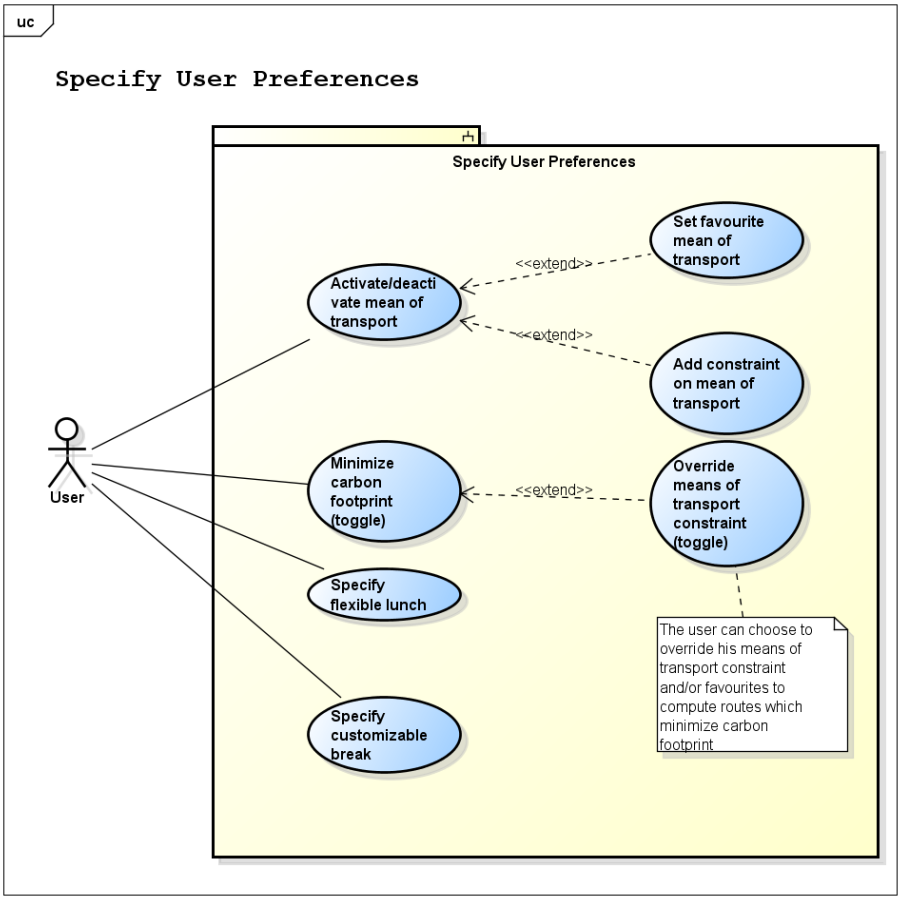

Figure 3: Use cases showing the user preferences and relative functonalities

3.6.10 Means activation/deactivation

Name	**Activate/deactivate mean of transport**
Actors	User
Entry conditions	The user is logged in and is in the user preferences page
Flow of events	• The user clicks on "Choose means of transport" button and he is redirected to a page containing a list of all possible means of transport; • The user unflags all the means of transport he does not intend to use. • The user clicks on "Done" button and is redirected to the user preferences page.
Exit conditions	The unflagged means of transport are removed from the possible means needed to compute a route
Exceptions	The user unselected every mean of transport, clicking "Done" button has no effect and an error message stating that at lest one mean of trasport has to be flagged.

Table 7: This is my one big table

3.7 Class Diagram

4 Alloy

4.1 Model

4.2 Result

immagine dei risultati

4.3 Worlds Generated

5 Effort Spent