Travlendar+ project Daverio Fiorillo

POLITECNICO
MILANO 1863

Requirement Analysis and Specification Document

Deliverable:	RASD
Title:	Requirement Analysis and Verification Document
Authors:	Daverio Fiorillo
Version:	1.0
Date:	29-October-2017
Download page:	https://github.com/FiorixF1/DaverioFiorillo.git
Copyright:	Copyright © 2017, Daverio Fiorillo – All rights reserved

Many endeavors require scheduling meetings at various locations all across a city or a region (e.g., around Lombardy), whether for work or personal reasons (e.g., meeting the CEO of a partner company, going to the gym, taking children to practice, etc.). The goal of this project is to create a calendar-based application that: (i) automatically computes and accounts for travel time between appointments to make sure that the user is not late for appointments; and (ii) supports the user in his/her travels, for example by identifying the best mobility option (e.g., use the train from A to B and then the metro to C), buying public transportation tickets, or by locating the nearest bike of a bike sharing system. Users can create meetings, and when meetings are created at locations that are unreachable in the allotted time, a warning is created. As mentioned, the application should also suggest travel means depending on the appointment (e.g., perhaps you bike to the office in the morning, but the bus is a better choice between a pair of afternoon meetings, and a car – either personal, or of a car-sharing system – is best to take children to practice) and the day (e.g., the app should suggest that you leave your home via car in the morning because meetings during a strike day will not be doable via public transportation; it could also take into account the weather forecast, and avoid biking during rainy days). Travlendar+ should allow users to define various kinds of user preferences. It should support a multitude of travel means, including walking, biking (own or shared), public transportation (including taxis), driving (own or shared), etc. A particular user may globally activate or deactivate each travel means (e.g., a user who cannot drive would deactivate driving). A user should also be able to provide reasonable constraints on different travel means (e.g., walking distances should be less than a given distance, or public transportation should not be used after a given time of day). Users should also be able, if they wish to, to select combinations of transportation means that minimize carbon footprint. Additional features could also be envisioned, for instance allowing a user to specify a flexible "lunch". For instance, a user could be able to specify that lunch must be possible every day between 11:30- 2:30, and it must be at least half an hour long, but the specific timing is flexible. The app would then be sure to reserve at least 30 minutes for lunch each day. Similarly, other types of breaks might be scheduled in a customizable way

Contents

List of Figures

1 Introduction

1.1 Purpose

The following document is the Requirement Analysis and Specification Document (RASD) for the formalization and description of all the needed features, constraints and recommendations that will constitute the proposed system.

The paper will focus on the elicitation and analysis of all functional and nonfunctional requirements, also verified by automated logic verification software (Alloy Analyzer [MIT]) and supported by UML schemes and scenarios. The provided document will also include a concise analysis of the environment and it tries to clarify all the interaction among system parts and external world.

The target audiences of this document are all the stakeholders involved in the development of the system and it can be used as contractual base for the project.

1.2 Scope

1.2.1 System description

The system that would be developed is a calendar-based application that provides support for planning of appointments, automating the travel arrangement process according to the user preferences. The system will be able to plan travel routes choosing transport options among public and private means, including trains, trams, taxis, bicycles, bike and car sharing, owned automobile and others. Therefore, the system will provide a set of possible route options, trying to minimize travel times, route length, carbon footprint or number of changes.

Also, it will have to recollect travel informations on public transportations and travelling conditions from different external sources. Beside, the application will give the possibility to users to register themselves, allowing them to access a series of extra features, including the possibility of setting personal preferences and backup personal agenda and routes.

User preferences include the possibility to set travel pauses, flexible launch (30 minute of travel pause between 11.30-14.30), enabling and disabling certains types of transportation means in a specific period of the day or anytime and in addition it will give the possibility to user to create profiles of preferences and link them to single appointments) Finally, system should be capable to interacting with any type of external public transportation service, although it will focus on make available them in the city of Milan.

1.2.2 World Phenomena

Many public transport service companies offers the possibility to obtains informations on timetables, routes, stations and status of services in aggregate way providing public Application Public Interfaces (APIs). Thus it is guaranteed the theoretical possibility to make queries in any moment and get the latest valid informations on transports.

Others doesn't offer the same possibilities; for instance Trenitalia hasn't yet a public API, but there are different opensource projects that allow to fill the gap.

Moreover, a lot of web mapping services makes available public APIs too. In this case it is possible to get GPS location from an address or vice-versa, or update a portion of map and even process a route given a starting point and a destination. All the required supported companies provide aforementioned services.

1.2.3 Goals

G1 Allow the user to add an appointment

- [G1.1] The user can add the date of the appointment through a calendar
- [G1.2] The user can select the location through a map

[G1.3] The appointment must be processed by the system

G2 - Provide a route to the user for reaching the appointments

[G2.1] The user must reach on time his/her appointments

[G2.2] The user can choose the starting location and time of the route

[G2.3] Generate routes according to the preferences of the user

[G2.4] The application provides a route for each objective, minimizing each of these attributes: length, duration, number of changes, carbon footprint

[G2.5] Always provide a route

[G2.6] During strike days, public transportation must not be available

[G2.7] Report unfavorable weather conditions

[G2.8] Update unfavorable weather conditions

[G2.9] The user can save one route among the shown ones

G3 Allow the user to sign up into the application

[G3.1] The registration must allow the univocal recognition of the user

G4 Allow the user to log in

[G4.1] The system allow the login through e-mail and password

[G4.2] The application allow the login through telephone number

G5 Allow the user to add his own preferences

[G5.1] The user must be logged

[G5.2] The preferences are synchronized between the application and the database

[G5.3] The user can chose the available transport means

[G5.4] The user can add a priority to the means of transport

[G5.5] The user can add the maximum length of routes on foot or by bicycle

[G5.6] For each vehicle the user can choose a time slot of validity

[G5.7] The user can set Flexible Lunch

[G5.8] The user can add breaks for fixed moments of the day

[G5.9] The user can add a private car or bicycle

[G5.10] The user can organize his preferences in “Preferences Profiles"

G6 Allow the user to manage his account

[G6.1] The user must be able to remove appointments and routes

[G6.2] The user must be able to modify appointments and routes

[G6.3] The user must be able to delete his account

G7 Allow the user to report events and disservices

[G7.1] The user can report strikes using the application

[G7.2] The user can report faults, malfunctions and suggestions

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- API(s) : Application Program Interface(s)
- RASD : Requirement Analysis Specification Document
- Appointment: location the user has to reach at a fixed date and time
- Route: journey between two appointments, it is composed by a series of paths
- Path: part of a route traversed with a specific mean of transport

1.3.2 Acronyms

- UI: User Interface

1.3.3 Abbreviations

- MA (Mobile Application): part of the system
- PT: Public Transportation

1.4 Revision history

1.5 References

- Specification Document assigned: `"Mandatory_Project_Assignments.pdf"`
- Software Requirements Specification Guidelines : IEEE 830-1998
- Alloy Website: http://alloy.mit.edu/alloy/

1.6 Document Structure

The document is composed of six different parts.

1. The introduction has the objective to support the reader into the reading of the document providing a brief description of the problem and the actual technological landscape description. It also include a list of definition, abbreviations and acronyms commonly employed through the rest of the document.
2. The second part offers an overall description of the system, then focused on the boundaries that divide the system from world, especially inspecting all interactions on shared phenomena. In addition, it is provided a short description of users. FInally, it is listed and described a set of core functionalities that will be supported and assumptions.
3. The third part provides a more complex and detailed description of functionalities, also in terms of requirement. This part is addressed specifically to development team
4. This section provides an Alloy model for verification of goals through a formal analysis of requirements and domain assumption. This is beneficial in assuring the consistency of the model. It also provide a representation of the world and system.
5. The fifth part provides the effort spent in term of time involved into the project by the authors. In the last part, there are references to external sources.

2 Overall Description

2.1 Product perspective

2.1.1 User interfaces

- A new user should be able to make an account, without training, in less than 10 minutes
- The number of steps requested to user in "appointment creation" should be less than 5
- User should be allowed to access system functions from mobile devices such as smart-phone and tablets
- User should be allowed to access system functionalities from PC
- UI for mobile devices should adapt to different screen sizes (from 5 to 11 inch)
- There shouldn't be need of training for new user to learn all application functionalities offered by UI

2.1.2 System interfaces

System have to communicate with external services, from public transportation means to map services. In order to guarantee future support to new services, also outside Milan area, it ought to be useful a modular, flexible but also extendible approach on the building of external interface. In the following part of the paragraph it will be shown a list of logic operation that should be accomplished by external services APIs in order to be interfaced and employed by the system.

- Train services and trams

  Looking for station list

  System request: station list

  Service response: list of station with locations

  Looking for train/tram

  System request: departure and arrival stations, departure time

  list of possible rides, cost of travel, times of leaving and arrival (at least 1 list's element)

- Taxi

  Looking for taxi

  System request: departure position and time

  Service response: disponibility and (hourly or journey cost)

- Bike sharing

  Looking for service parking spot

  System request: spaces

  Service response: list of spaces with available bicycles

  Looking for fees

  System request: fees

  Service response: hourly fee

- Map services

  System should be able to retrieve the nearest address given GPS coordinates and vice-versa by these services.

  Map services should be able to generate a route between two given address on system request.

  System should be allowed to download a portion of map.

- Weather forecast service

  Request forecast

  System request: forecast for a specific location and time

  Service response: weather conditions and temperature for requested locality and time

2.1.3 Software interfaces

In order to ship a complete functional software, at least for the city of Milan, a list of required supported external APIs is provided in this paragraph.

- Trenitalia

  Name: “Informazioni-Treni-Italiani”

  Description: unofficial API for Trenitalia, further details on source

  Version: testing

  Source: https://github.com/Razorphyn/Informazioni-Treni-Italiani

- ATM

  Name: “localizzazione delle fermate delle linee urbane di superficie”

  Description: list of metro stops and hours in data-sheet version, regularly updated

  Version: 29/12/2015

  Source: http://dati.comune.milano.it/

- Bike Sharing

  Name: “Mobilità: localizzazione delle rastrelliere per il bike sharing”

  Description: list of bikesharing areas in datasheet version, regularly updated

  Version: 05/12/2016

  Source: http://dati.comune.milano.it/

- Taxi

  Name: “appTaxi | Developers”

  Description: restricted API to book or calculate prices for a ride

  Version: rolling

  Source: www.apptaxi.it/developers/

- Weather

  Name: “Open Weather Map API”

  Description: open API to get weather conditions for a location

  Version: rolling

  Source: https://openweathermap.org/api

2.2 Product functions

It has been selected a set of functionalities, on one hand in order to provide to user a service that focus on easy of use, automation and simplification of user processes and interactions with the system, on the other hand allowing him to freely compose complex schemes of constraints in the choice of travel routes.

2.2.1 Preference profile creation

A logged user should be able to create some types of constraints (flexible launch, disabilitation of transport means, set pauses from transport and maximum travelling distance by types of transport mean). He should also be allowed to create many different preferences profiles, that are logical containers for sets of user preferences. These profiles can be applied to single appointments personalizing the research of a routes. Preferences with unspecified profiles belongs to "default" profile and they will ever be considered as constraint for route generation.

2.2.2 Appointment insertion

A user can register an appointment providing a description, a date, a time of begin and end. Beside, a registered user can optionally add a list of preferences profiles. The appointment shouldn't conflict with others of the same user, so if it happens, it will be rejected. Accepted appointment have to be processed, and route options prompted to user.

2.2.3 Appointment modification

A user can modify any detail of a previous inserted appointment. If the user is logged into the system, all modifications have to be synchronized with user account. Finally, system should provides new routes for reaching the modified appointment, deleting previous saved routed linked with the old one.

2.2.4 Change travel route

A user can choose to change a saved route. In this case, system will delete user current selected saved route and it will provides new routes to user.

2.2.5 Route choice

When requested, the system have to generate routes to reach appointment according to registered user preferences profiles and the previous appointment. System will provide a route that will minimize distance, another one minimizing duration and others minimizing carbon footprint or number of changes.

2.2.6 User Registration

User can choose to register an account into the system through MA indicating a valid email address, a secure password, name, surname and fiscal code.

2.2.7 User login

A registered user can log in using phone number and sms verification process or by email and password.

2.3 User characteristics

It's possible to divide application users into two groups:

- Unregistered

  He can access some app features, including appointment registration and route choices

- Registered

  He can access all the feature set that system can provide, so all the functionality provided to unregistered user, plus the possibility to set profiles of preferences, synchronization and backup of user data and preferences. This user can also receive weather forecast update.

The user of this system is not required to have IT skills.

2.4 Assumptions

D1 The timetables of the public transport means are retrieved by external API and are reliable.

D2 The route between two points is calculated by external map and routing services, they provide correct information.

D3 The GPS gives the correct position with an accuracy of few meters.

D4 The registered users are recognized by their email address, which is unique.

D5 If the system needs to send an SMS to a user, he will receive it.

D6 A route between two locations is composed by some paths, each of them is travelled by a specific mean of transport.

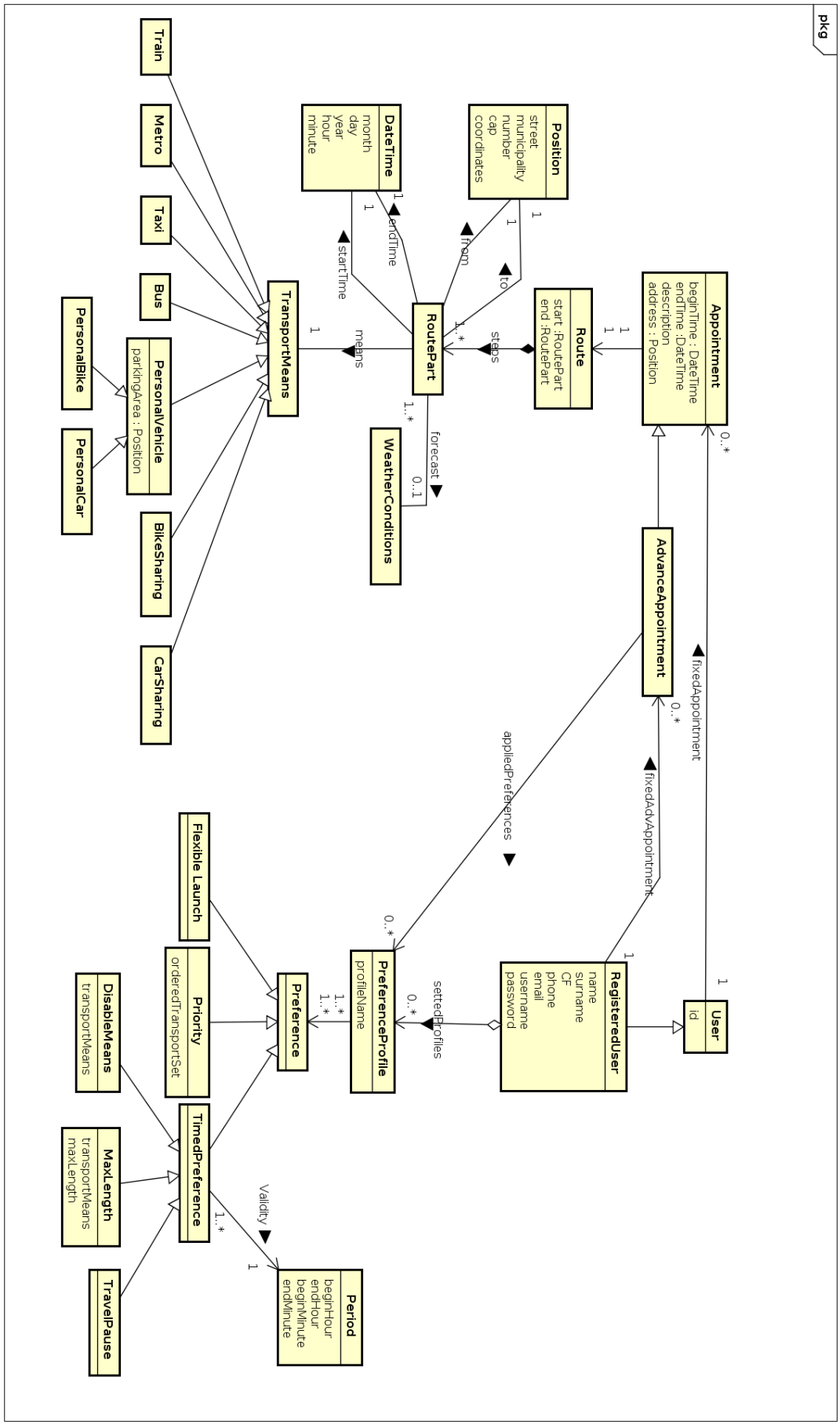

Figure 1: UML class diagram perspective

3 Specific Requirements

3.1 Hardware Constraint

- iOS or Android smartphones
- PC/Mac support
- WiFi and web connection
- GPS

3.2 Functional Requirements

G1 Allow the user to add an appointment

[G1.1] The user can add the date of the appointment through a calendar

[R1.1.1]The application must provide the user with a calendar in order to choose the date

[G1.2] The user can select the location through a map

[R1.2.1]The application must provide a map in order to obtain the GPS coordinates

[G1.3] The appointment must be processed by the system

[R1.3.1]The application accepts only appointments with at least date, starting time, ending time, location and description with no overlapping with other appointments

[R1.3.2]The appointment is processed and registered if it doesn't overlap with other appointments of the logged user

[R1.3.3]The location is solved completing it with GPS coordinates and address using external map services

[R1.3.4]The application must notify the user about possible errors that prevents the processing of the appointment to the user

G2 - Provide a route to the user for reaching the appointments

[G2.1] The user must reach on time his/her appointments

[R2.1.1]The arrival time must precede the starting time of the appointment

[R2.1.2]The starting time of the route must be after the ending time of the previous appointment

[R2.1.3]The application must send a notification to the user to remind him about the next travel

[G2.2] The user can choose the starting location and time of the route

[R2.2.1]From the location and the ending time of the previous appointment (if it exists)

[R2.2.2]From a personalized location and time

[R2.2.3]From the current location and time

[G2.3] Generate routes according to the preferences of the user

[R2.3.1]All global preferences of the user must be satisfied

[R2.3.2]The user can add one preference profile for the current appointment

[R2.3.3]The preferences provided in the preference profiles are constraints connected by logical AND

[G2.4] The application provides a route for each objective, minimizing each of these attributes: length, duration, number of changes, carbon footprint

[R2.4.1]Each route provided must indicate starting and ending time, IDs of public transportation to use, distance, carbon footprint, paths

[R2.4.2]The application must remind the user about the start of the route

[R2.4.3]The user can order the result according to

1. Length
2. Duration
3. Number of changes
4. Carbon footprint
5. The public transport timetables are obtained by contacting external services

[R2.4.4]The available means of transport are

1. Private car
2. Bicycle
3. On foot
4. Public transportation
5. Bus
6. Tram
7. Underground
8. Train
9. Taxi
10. Car sharing
11. Bike sharing

[G2.5] Always provide a route

[R2.5.1]If there are no routes that satisfy all the requirements, search again allowing an arrival time after the starting time of the appointment

[R2.5.2]Report the arrival delay with a warning

[G2.6] During strike days, public transportation must not be available

[R2.6.1]If a society emerges to be striking, its services must not be available for the calculation of the route

[G2.7] Report unfavorable weather conditions

[R2.7.1]Add a weather warning to the route if part of it is on foot or bicycle in locations or hours in which unfavorable weather conditions are expected during the calculation of the route

[R2.7.2]Unfavorable weather conditions are rainfalls or temperature under 3 C

[R2.7.3]Weather forecasts are requested through external services

[G2.8] Update unfavorable weather conditions

[R2.8.1]The user must be logged

[R2.8.2]Update the weather conditions of the saved routes every 6 hours and notify unfavorable weather conditions on routes with parts on foot or bicycle if not already done previously

[G2.9] The user can save one route among the shown ones

[R2.9.1]The user can choose anyone of the provided routes

[R2.9.2]The choice must be saved in the database if the user is logged

G3 Allow the user to sign up into the application

[G3.1] The registration must allow the univocal recognition of the user

[R3.1.1]The registration is valid if it contains at least First Name, Family Name, date of birth, fiscal code, telephone number, e-mail address, valid password

[R3.1.2]The fiscal code is valid if it agrees with name, family name and date of birth

[R3.1.3]The e-mail address is valid if it is not already taken in the database

[R3.1.4]The password is valid if it contains at least 6 letters

[R3.1.5]The system can accept only valid registrations

G4 Allow the user to log in

[G4.1] The system allow the login through e-mail and password

[R4.1.1]The user can request for the access into the application using e-mail and password

[R4.1.2]The access is guaranteed only if e-mail and password prove to be linked to an account

[R4.1.3]The application must notify the user about the outcome of the login

[G4.2] The application allow the login through telephone number

[R4.2.1]The user can request for the access using the telephone number

[R4.2.2]The system must be able to send an SMS to the provided number with a verification code of 4 digits

[R4.2.3]The user can complete the login if the verification code inserted by the user matches the one generated by the system

G5 Allow the user to add his own preferences

[G5.1] The user must be logged

[G5.2] The preferences are synchronized between the application and the database

[R5.2.1]Each modification to the user settings must be reflected in the update of the database

[R5.2.2]The preferences can be changed only if the user is logged

[G5.3] The user can chose the available transport means

[R5.3.1]The subsequent routes can use only the vehicles marked as available in the preferences

[G5.4] The user can add a priority to the means of transport

[R5.4.1]Each mean of transport can have a priority among low, normal and high

[R5.4.2]While calculating the route, if more vehicles are available for the same path, the one with highest priority is chosen

[G5.5] The user can add the maximum length of routes on foot or by bicycle

[R5.5.1]Distances can be expressed in meters or kilometers

[G5.6] For each vehicle the user can choose a time slot of validity

[R5.6.1]The user can choose one or more time slots of validity for a given vehicle with the possibility of scheduling on a weekly basis and with accuracy of minutes

[G5.7] The user can set Flexible Lunch

[R5.7.1]Given the time slot for the lunch break and its minimum duration, the application must guarantee that, for at least the specified duration, the user is not traveling during that time slot

[R5.7.2]The minimum duration of a lunch break must not exceed the duration of the chosen time slot

[G5.8] The user can add breaks for fixed moments of the day

[R5.8.1]The insertion of the time slot by the user is mandatory

[R5.8.2]During the time slot set by a break, no scheduled journey must be in progress

[G5.9] The user can add a private car or bicycle

[R5.9.1]It is needed to indicate the habitual parking zone of the vehicle.

[G5.10] The user can organize his preferences in "Preferences Profiles"

[R5.10.1]The user can include a preference in one or more preferences profiles

[R5.10.2]If no profile is specified, the preference has global validity

G6 Allow the user to manage his account

[G6.1] The user must be able to remove appointments and routes

[R6.1.1]After removing a appointment, the previous and following routes are removed as well

[G6.2] The user must be able to modify appointments and routes

[R6.2.1]In case of modification of a route, the application will provide the user with a new list of newly generated routes to choose

[R6.2.2]In case of modification of the starting or ending time or the location of a appointment, the eventually present previous and subsequent routes will be removed

[G6.3] The user must be able to delete his account

[R6.3.1]The removal of the account implies that the system deletes the credentials and all personal data and information of the user

G7 Allow the user to report events and disservices

[G7.1] The user can report strikes using the application

[R7.1.1]The user must provide the striking society and report the day of the strike and the starting and ending time

[G7.2] The user can report faults, malfunctions and suggestions

[R7.2.1]The user must indicate the reason of his report with a textual message

3.3 Software System Attributes

3.3.1 Performance

The system must answer rapidly to the requests of the users while searching for timetables and calculating the routes. The performance of the system depends also on the response time of the external services.

3.3.2 Reliability

The system must be available 24/7. However also in this case the availability of the service depends on the one of external services. The lack of availability of the former can limit the functionalities of the main application

3.3.3 Security

Some personal data are stored in the database, so it is necessary to guarantee security over these data.

3.3.4 Scalability

The system can be expanded with new external services. It must be designed in such a way that makes possible modifying or adding other services for accessing the timetables, the maps and the weather forecast.

3.3.5 Accuracy

The timetables and the estimated length and duration of the routes must be as precise as possible. Also the weather forecast must be updated regularly to provide reliable information.

3.4 More Non Functional Requirements

3.4.1 Licensing

Front End application will be eventually distributed under Freeware License

3.5 Scenarios

3.5.1 Scenario 1

Bob is looking for a service that allows him to keep track of all his work meetings around Milan. Surfing on web, he finds Travlendar+, so he decides to use that for his schedules. Downloaded the app, from the homepage, he requests the list of his registered appointments, actually empty. Then, he add a new appointment providing the date and time of beginning and end of meeting, finally, a brief description. He didn't choose to compile the departure and hour position, so now he can choose five different options to reach the meeting from his actual position. He decide to save the route with lower travel time and waits for the suggested beginning of travel.

3.5.2 Scenario 2

Alice has a very busy life, but bad memory too, so she has adopted to use Travleandar+ everyday to reminds her all her work and private appointments. Now she decides to make an account to keep synced appointments among all her devices, and have always a backup in case she loses a device. Therefore, she selects the option that permits to create a new account, she fills all fields required to her, including name, surname, FC, email, new username and password. Finally, she logged in from different devices finding her list of appointment up to date.

3.5.3 Scenario 3

Tracy is a registered user of Travlendar+. She wants to personalize the choice of routes in order to fulfill her needs and preferences. She is already logged in Travlendar+, so she can choose to view and modify her preference profiles through the specific function. She adds a new preference profile providing a name for it, then she creates the flexible launch preference and she sets also a travel pause between 7-8 pm. Finally, she added the new preferences to the just created preference profile. Now she can apply the profile to all new and already setted appointment during their creation or modification.

3.5.4 Scenario 4

Bob has just been informed about a change in the afternoon meeting. He has already added this meeting appointment in Travlendar+ sometime before. By the appointment view, he chooses the one he wants modified, and he changes only the beginning hour, anticipating that by half an hour, according to new schedule. Now he is asked to choose another among others routes to reach the modified appointment in time.

3.5.5 Scenario 5

Luna is a environmentally conscious person that always tries to help the nature. So she uses Travlendar+ with the objective to minimize her travel carbon footprint. For the last inserted evening appointment, she accidentally selects a route that implies a lot of CO2 emissions, so she wants to change that. She navigates her appointment list, selects the evening appointment and she can select another route among different choices. She finally saves the route that minimize carbon footprint.

3.5.6 Scenario 6

Alice is afraid of travel alone by train at night because of self security concerns. Therefore she has decided to always avoid trains between 22.00 and 5.00. On Travlendar+, she adds to "default" preference profile a specific rule that permits her to disable a transport means for a specific period of the day. As far the new rule is linked to "default" is always considered in route generation, so she will be never asked to travel by train at night.

3.5.7 Scenario 7

Tracy has just been notified by Travlendar+ about icy temperatures on the following day. So she decides to change one route of the following day avoiding walking and bicycles. Through the notification, she obtains a list of her routes with walking or bike parts with bad forecast or freezing temperatures, and for each of them, she can choose an alternative path.

3.5.8 Scenario 8

Bob is the owner of a private car that he would like to use during his travels. Since he is already registered and logged in, he can add, as global preference (to "default" profile) his car, indicating where it is normally parked. By this moment, every new generated route for an appointment can consider the employment of own registered means.

3.6 Use case description and diagram

Actors	User
Input Conditions	There are no input conditions.
Events Flow	- The user is on the main page of the application and click the "Sign up" button. - He provides the required data for the registration and clicks "Confirm". - The application confirms the registration.
Output Conditions	The user is on the new personal page.
Exceptions	- The request cannot be processed due to errors in input - The fiscal code is not compatible with the other fields - The e-mail address is already taken by another user. - The password is less than 6 characters long. In all cases, the system refuses the registration and asks to provide correct data.

Figure 2: Use case - User Registration

Actors	User
Input Conditions	There are no input conditions.
Events Flow	- The user is on the main page of the application and click the "Log in" button. - He enters the request data: e-mail address and password. - The application processes the data.
Output Conditions	The user is redirected on the personal page.
Exceptions	- The user doesn't have an account - The e-mail address is not associated to any account - The entered password is wrong In these cases the system prevents the log in and asks for correct data. - The user doesn't remember his credentials. In this case, the user can select the "Forgot credential" option and he will receive an SMS to his telephone number with a verification code.

Figure 3: Use case - User Login

Actors	User Map service
Input Conditions	The user can be logged.
Events Flow	- The user is on the personal page and clicks the "Add appointment" button. - He provides the information of the appointment. - He inserts the date through a calendar. - He inserts the starting and ending time through a clock widget. - He inserts the location through an address or an interactive map. - He eventually adds a description. - The user confirms the form. - The application eventually solves the address using map services. - The application checks the validity of the parameters.
Output Conditions	The appointment has been registered in the calendar of the user.
Exceptions	- There is an overlapping between the timeslot of the appointment and an existing appointment or route. The system notifies the user of the presence of an overlapping and asks to enter a different date or time. - The address doesn't exist. The system notifies the user when the inserted address was not found and asks to insert a different address.

Figure 4: Use case - User adds an Appointment

Actors	User Map service Public transport service Weather service
Input Conditions	The user can be logged and has at least one appointment on the calendar.
Events Flow	- The user is on the personal page and choose an appointment from the calendar. - On the page of the appointment, the user selects the "Calculate route" option. - The system asks from where and when the user wants to start the route towards the location of the appointment: from the previous appointment (if it exists), from here and now, from a personalized location and time. - The user chooses one of the three options. - In case of the personalized location and time, the system shows a new page where the user can insert the starting location and time. A map service will solve the address when confirming. - After that, if the user is logged with an account, he chooses a preferences profile for this route. He can choose the global profile, a specific profile or a new profile defined at the moment. - The system calculates the possible routes, using external services for public transport timetables and weather conditions. - For each calculated route, the system gives a warning if the user can not be on time at the appointment or if there are unfavorable weather conditions during a path on foot or by bicycle. - The user chooses one route.
Output Conditions	The chosen route has been registered in the calendar of the user.
Exceptions	- The route overlaps with the appointment. In this case, the system warns the user. If starting location and time are manually inserted, an overlap can happen also with appointments that are not the one the user wants to reach, or even other routes.. - The address doesn't exist. The system notifies the user when the starting address was not found and asks to insert a different address. - No route satisfies the constraints. The system shows an empty list of routes explaining that no route was found satisfying all constraints.

Figure 5: Use case - User adds a route to an appointment

Actors	User Map service
Input Conditions	The user can be logged and has at least one appointment on the calendar.
Events Flow	- The user is on the personal page and selects an appointment on the calendar. - A summary of the data of the appointment appears on the screen. - The user click on the "Change" option. - He inserts new data: location, starting and ending time, description. - The user confirms the form. - If the location has changed, the application tries to solve the address using map services. - If the location or the time are changed, the application warns the user and asks if he is sure to make these changes. - If the user confirms the changes, all routes using that appointment as starting or ending point are deleted. - The user confirms his decision. - The application saves the new parameters.
Output Conditions	The appointment has been modified..
Exceptions	- The new location or time overlaps with an existing appointment or route. The system notifies the user of the presence of an overlapping and asks to enter a different date or time. - The address doesn't exist. The system notifies the user when the inserted address was not found and asks to insert a different address.

Figure 6: Use case - User changes appointment details

Actors	User
Input Conditions	The user can be logged and has at least one appointment on the calendar.
Events Flow	- The user is on the personal page and selects an appointment on the calendar. - A summary of the data of the appointment appears on the screen. - The user click on the "Delete" option. - The application asks the user if he is sure to delete the appointment. This will also delete all routes using the appointment as starting or ending point. - The user confirms his decision.
Output Conditions	The appointment and its routes has been removed.
Exceptions	There are no exceptions in this use case.

Figure 7: Use case - User delete appointment

Actors	User Map service Public transport service Weather service
Input Conditions	The user can be logged and has at least one appointment with a route.
Events Flow	- The user is on the personal page and selects an appointment on the calendar. - The user click on the "Change route" option. - The subsequent actions are the same for creating a new route.
Output Conditions	The new route has been registered in the calendar of the user.
Exceptions	- The route overlaps with the appointment. In this case, the system warns the user. If starting location and time are manually inserted, an overlap can happen also with appointments that are not the one the user wants to reach, or even other routes.. - The address doesn't exist. The system notifies the user when the starting address was not found and asks to insert a different address. - No route satisfies the constraints. The system shows an empty list of routes explaining that no route was found satisfying all constraints.

Figure 8: Use case - User modifies a route

Actors	User
Input Conditions	The user can be logged and has at least one appointment with a route.
Events Flow	- The user is on the personal page and selects an appointment on the calendar. - A summary of the data of the appointment appears on the screen. - The user click on the "Delete route" option. - The application asks the user if he is sure to delete the route. - The user confirms his decision.
Output Conditions	The route has been removed.
Exceptions	There are no exceptions in this use case.

Figure 9: Use case - User deletes a route

Actors	User
Input Conditions	There are no input conditions.
Events Flow	- The user is on the personal page and choose the option "Report" from the settings. - The system asks to the user if he wants to report a strike or a disservice/suggestion. - If the user wants to report a disservice or a suggestion, a generic text area appears where he can write the report. - Else a form will appear to enter the date, starting time and ending time of the strike, with the name of the striking society. An additional text area can be used for further information. - The user confirms the report.
Output Conditions	The report has been sent.
Exceptions	There are no exceptions in this use case.

Figure 10: Use case - The user reports events and disservices

Actors	User
Input Conditions	The user is logged.
Events Flow	- The user is on the personal page and choose the option "Add profile preferences" from the settings. - The application shows a form to fill. - The user gives the following information: a name for the profile, the list of usable vehicles, a level of priority for each selected vehicle (low, normal, high), a timeslot of availability for each selected vehicle, the maximum length he can travel on foot or by bicycle, the presence of flexible lunch with timeslot and minimum duration, any further breaks with their timeslot. - The user confirms the report.
Output Conditions	The profile preference has been saved.
Exceptions	There are no exceptions in this use case.

Figure 11: Use case - The user adds a profile preference

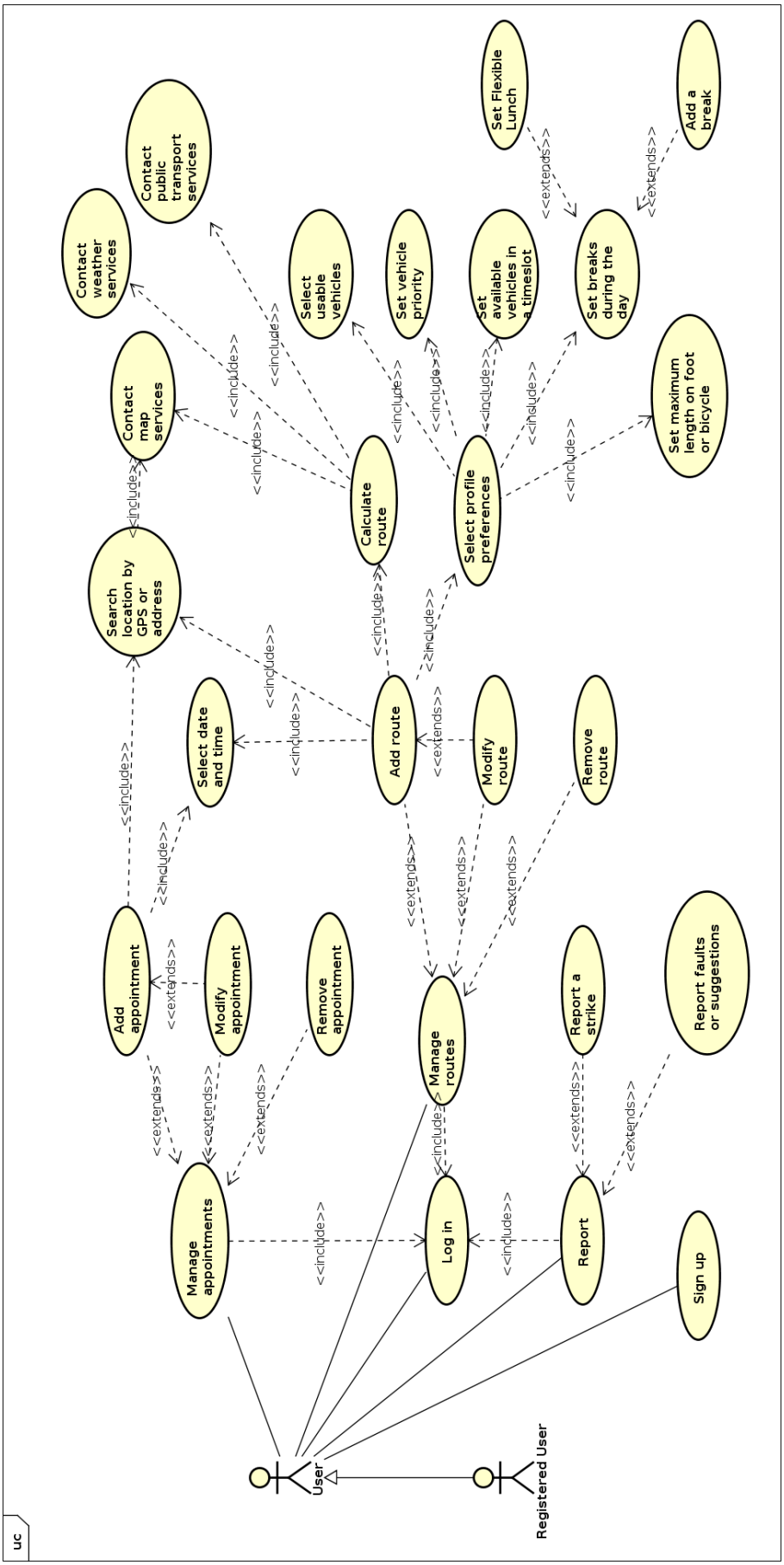

Figure 12: Use case diagram

4 Formal Analysis Using Alloy

4.1 Signatures

```
//Signatures

sig User {
   id : Int,
   fixedAppointments : set Appointment
}

sig RegisteredUser extends User{
   username : one Username,
   password : one Password,
   email : one Email,
   name: one Name,
   surname : one Surname,
   cf : one CF,
   fixedAdvAppointments : set AdvancedAppointment,
   preferenceProfiles : set PreferenceProfile
}

sig Appointment{
   beginTime : one DateTime,
   endTime : one DateTime,
   description : one string,
   address : Position,
   route :  set Route
}

sig AdvancedAppointment extends Appointment{
   appliedPreferences : set PreferenceProfile,
}

sig Route {
   steps : some RoutePart,
   start : one RoutePart,
   end : one RoutePart
}

sig RoutePart {
   from : Position,
   to : Position,
   startTime : DateTime,
   endTime : DateTime,
   forecast : lone WeatherCondition,
   travelMeans : TravelMeans
}

sig Position {
   street : string,
   municipality : string,
   number : Int,
   cap : CAP,
   coordinates : Coord
}

sig PreferenceProfile{
   preferences : set Preference,
   name : string
}

one sig DefaultProfile extends PreferenceProfile {}
```

Figure 13: Alloy signatures 1/2

```
//Other Sig
sig Username{}
sig Password{}
sig Email{}
sig Name{}
sig Surname{}
sig CF {}
sig CAP {}
sig string{}
sig Coord{}
sig WeatherCondition{}
sig Preference {}
sig TravelMeans {}


sig DateTime{
  month : Int,
  day : Int,
  year : Int,
  hour : Int,
  minute : Int
}
```

Figure 14: Alloy signatures 2/2

4.2 Facts

```
//All normal appointment are linked to normal users, AdvancedAppointments to RegisteredUsers
fact AppointmentAndAdvancedAppointment{
   all a: Appointment, ad: AdvancedAppointment | ((a in (Appointment - AdvancedAppointment)) implies a in U
      && ad in RegisteredUser.fixedAdvAppointments
}

//Username, Email, CF for RegisteredUsers are keys
fact uniqueIdUsernameEmailCF{
   all u1, u2 : RegisteredUser | u1 != u2 implies
      (u1.id != u2.id && u1.username != u2.username && u1.cf != u2.cf && u1.email != u2.email)
}

//There aren't Usernames, Passwords, Email, Name, Surname, CF not linked to anyone
fact noOrphanUsernamePasswordEmailNameSurnameCF{
   all us : Username, c: CF, e : Email, p:Password, n:Name, s:Surname |
      us in RegisteredUser.username && c in RegisteredUser.cf && e in RegisteredUser.email
      && p in RegisteredUser.password && n in RegisteredUser.name && s in RegisteredUser.surname
}

//Constrains on Calendar Data
fact DateTimeCalendarConstrains{
   all dt : DateTime | dt.month <= Int[12] && dt.day <= 31 && dt.hour < 24 && dt.minute <60
      && dt.month > 0 && dt.day > 0 && dt.hour >= 0 && dt.minute >= 0
      && dt.year = 0 //Added to reduce scope and complexity
}

//Coords and WeatherCondition always linked to Position and RoutePart
fact noOrphanWeatherConditionCoord{
   all w: WeatherCondition, c : Coord | c in Position.coordinates && w in RoutePart.forecast
}

//Preferences are always linked to a PreferenceProfile as well as that is always linked to a RegisteredUser
fact PreferenceProfileConstraints{
   all p : Preference | p in PreferenceProfile.preferences
   all pp : PreferenceProfile | pp in RegisteredUser.preferenceProfiles
}

//WeatherCondition are guaranteed to RegisteredUser
fact WeatherConditionForRegisteredUser{
   all adst: AdvancedAppointment.route.(steps+start+end) | adst.forecast != none
}

//Coord are keys for Position
fact uniqueCoords{
   all p1, p2 : Position | p1 != p2 implies p1.coordinates != p2.coordinates
}

//RoutePart are not of null length or equal segments
fact lengthNotNull{
   all rp : RoutePart | rp.from != rp.to
   no disj rp, rp1 : RoutePart | rp.from = rp1.from and rp.to = rp1.to
}

//intemediate steps aren't start or end ones
fact {
   all r : Route | (no s : r.steps | r.start = s or r.end = s)
}
```

Figure 15: Alloy facts 1/2

```
//Route Parts are linked to form a Route
fact {
   all r : Route | (
      (lone s : r.steps | r.start.to = s.from)
      && (lone s : r.steps | r.end.from = s.to)
      && (no s : r.steps | r.start.from = s.to)
      && (no s : r.steps | r.end.to = s.from)
      && (all s : r.steps | s.to = r.end.from or one s1: r.steps | s.to = s1.from)
      && (all s : r.steps | s.from = r.start.to or one s1 : r.steps | s.from = s1.to)
      && (#r.steps = 0 implies (r.start = r.end or r.start.to= r.end.from) else r.start != r.end)
   )
}
```

Figure 16: Alloy facts 2/2

4.3 Predicates

```
//A new preference is added to DefaultProfile
pred createPreference(p : Preference, dp : DefaultProfile, dp' : DefaultProfile) {
   //prec
   !(p in dp.preferences)
   //postc
   all op : dp.preferences | op in dp'.preferences && p in dp'.preferences && #op = (#dp'.preferences -1)
}

//Add a preference to a profile (This implies the removal from DefaultProfile if present)
pred addPreference(p : Preference, dp : DefaultProfile, dp' : DefaultProfile, pp : PreferenceProfile, pp' : PreferenceProfile) {
   //prec
   !(p in pp.preferences)
   //postc
   all op : dp.preferences | op != p implies op in dp'.preferences else !(op in dp'.preferences)
   all np : pp.preferences | np in pp'.preferences && p in np
}

//Add preferenceProfile to AdvancedAppointment
pred addPreferenceProfileToAppointment(pp : PreferenceProfile, aa : AdvancedAppointment, aa' : AdvancedAppointment) {
   //prec
   !(pp in aa.appliedPreferences)
   //postc
   all pr : aa.appliedPreferences | pr in aa'.appliedPreferences &&  pp in aa'.appliedPreferences
}

//Add an Appointment to User (AdvancedAppointment <> RegisteredUser)
pred addAppointment(a : Appointment, u : User, u' : User) {
   //prec
   a in AdvancedAppointment iff u in RegisteredUser
   !( a in u.fixedAdvAppointments+u.fixedAdvAppointments)
   //postc
   a in AdvancedAppointment implies (all ad : u.fixedAdvAppointments | ad in u.fixedAdvAppointments && a in u.fixedAdvAppointments)
      else (all ab : u.fixedAppointments | ab in u.fixedAppointments && a in u.fixedAppointments)
}

//Remove an Appointment (AdvancedAppointment <> RegisteredUser)
pred removeAppointment(a : Appointment, u : User, u' : User) {
   //prec
   a in AdvancedAppointment iff u in RegisteredUser
   a in u.fixedAdvAppointments+u.fixedAdvAppointments
   //postc
   a in AdvancedAppointment implies
      (all ad : u.fixedAdvAppointments | ad != a implies ad in u'.fixedAdvAppointments else !(ad in u'.fixedAdvAppointments))
         else (all ba : u.fixedAppointments | ba != a implies ba in u'.fixedAppointments else !(ba in u'.fixedAppointments))
}
```

Figure 17: Alloy predicates

4.4 Results

Figure 18: Alloy consistency results

4.5 World

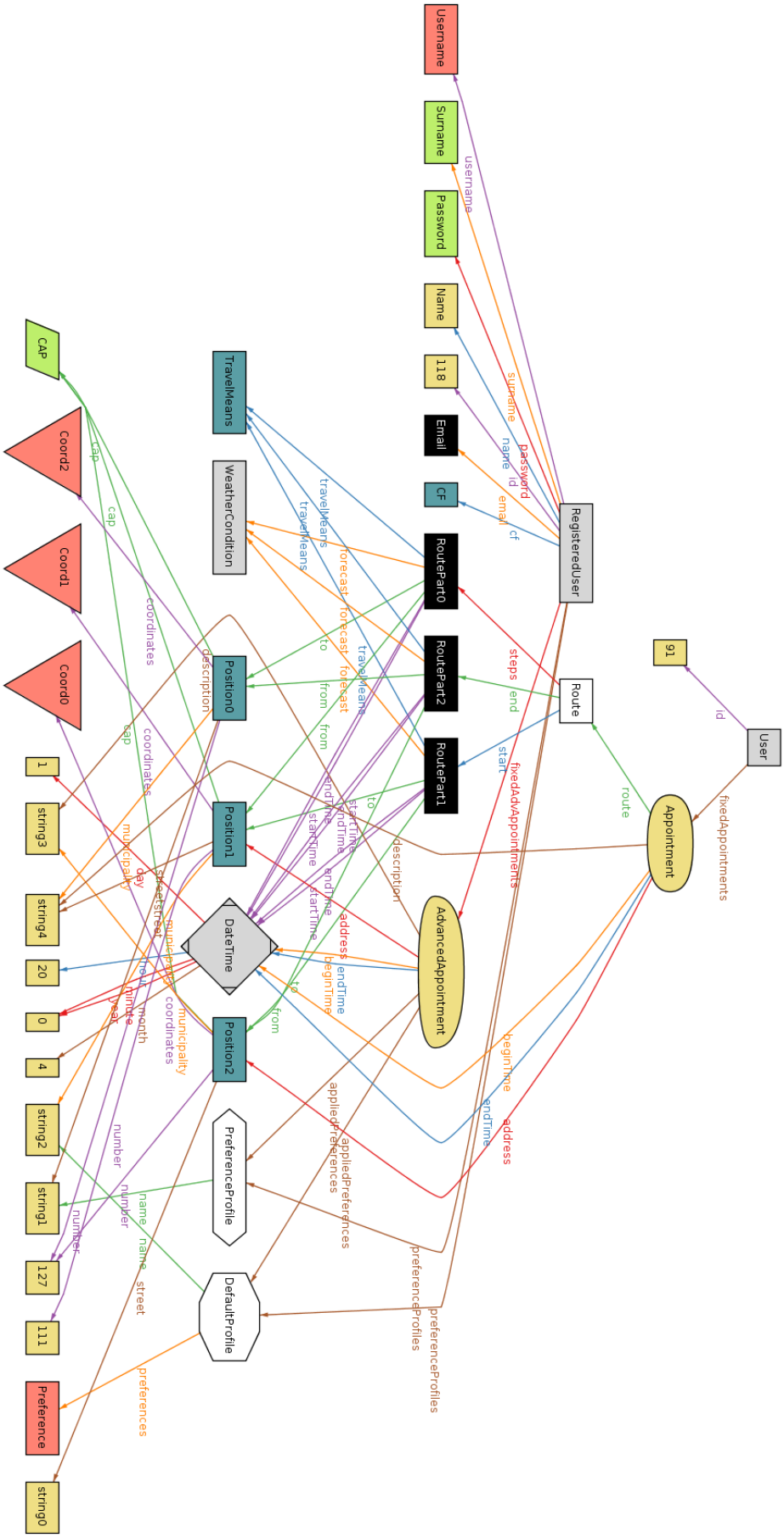

Figure 19: Alloy generated world

5 Effort Spent

5.1 Fiorillo

8/10/2017	45m
11/10/2017	45m
12/10/2017	1h20m
13/10/2017	1h
15/10/2017	2h
23/10/2017	1h15m
24/10/2017	30m
26/10/2017	2h15m
28/10/2017	30m
29/10/2017	3h
TOT	13h20m

5.2 Daverio

2/10/2017	30m
4/10/2017	40m
7/10/2017	30m
10/10/2017	2h20m
14/10/2017	1h40m
18/10/2017	3h
20/10/2017	45m
21/10/2017	1h20m
25/10/2017	1h45m
27/10/2017	7h45m
28/10/2017	9h30m
29/10/2017	16h45m
TOT	47h30m