Politecnico di Milano
AA 2017-2018

Computer Science & Engineering
Software Engineering Report

Travelendar+

RASD
Requirement Analysis and Specification Document

Mingju Li - 10574864/898045
Chang Lin - 10597034/894651

POLITECNICO DI MILANO

Directory

Ⅰ. Description of the problem

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** <u>*(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)*</u> **and the day** (<u>*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*</u>).

Travlendar+ should allow users to **define various kinds of user preferences**. It should **support a multitude of travel means**, <u>*including walking, biking (own or shared), public transportation (including taxis), driving (own or shared), etc.*</u> A particular user may **globally activate or deactivate each travel means** (<u>*e.g., a user who cannot drive would deactivate driving).*</u> A user should also be able to **provide reasonable constraints on different travel means** <u>*(e.g., walking distances should be less than a given distance, or public transportation should not be used after a given time of day).*</u> 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"**. <u>*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.*</u>

1. Introduction

1.1 Purpose

here we include the goals of the project

The purpose of this document is to give technical details advices about the architecture of Requirement Analysis and Specification Document(RASD) about Travelendar+ (From here on, to make it brief we call it **TVLD** in abbreviation). This document is addressed to developers who must implement the requirements and could be used as a contractual basis and aims to identify:

- The high-level architecture and boundaries of system
- The design patterns
- The main components and their interfaces provided one for another
- The Runtime behavior

1.2 Scope

here we include an analysis of the world and of the shared phenomena

1.2.1 Description

TVLD is designed to offer the user supports on the arrangements of timetable. TVLD is an information system based on mobile application and web application, which integrates information from different external systems and offer a highly optimized view of the information it obtains, using algorithms to calculate the best solution for a current situation.

It has one target group of users, people who would like to use this App to manage their time schedule. Of course, this group of users includes people involving different needs of time schedule managements, for example, a business manager may have different need from a house wife or a single student. Here are some scenarios of TVLD:

Scenario I: A CTO from a technology company just finishes the meal may have a meeting with the marking manager in company's meeting room at 15 PM, then go to school to take his children at 17 PM. All these stuffs are in different places, and if the travel time is considered, it is quite hard for him to make a good schedule.

Scenario II: A student is on his way to the university. He's course would end at 12:00 and then he gets an internship which will start at 14:30 on the other side of city. He would like to take a lunch break before goes to internship. While when he finished his lunch, he finds that there are not available Sharing bicycles nearby and it seems that it going to rain in 1 hour as well.

Besides the users, external companies are the potential targets of this App. To build this system we need the information of different external systems like (ATM, MoBike, …), and this stimulates the increase use of these third-party service as well. Thus, we could make profits based on this win-win strategy.

Generally speaking, this system will automatically compute the travel time between different appointments to help users to well manager their schedules, and provide necessary supports for them. If the location is reachable in the allotted time, the system will provide the most time-economic or cost-economic travel means depending on the user choice. Otherwise, if the location is unreachable, a warning will be alerted. The system will also take accounts of any other issues like weather, traffic jam or public strike. Ultimately, Travlendar+ should provide functions easy to use and allow user to set their own preference.

1.2.2 Goals

[G1] Allow users to set their time schedule including the location and time.

[G1.1] If the email is not verified, the user is considered as a visitor and only accesses to basic service of this APP

[G1.2] If the meeting is unreachable in the allotted time, a warning should be alerted.

[G1.3] If the meeting is reachable in the allotted time, it is created successfully in the meeting list.

[G1.4] Allow a user to edit the location and the allotted time, or even cancel the appointment.

[G2] Allow the users to define their preference on the traffic method.

[G2.1] Allow the user to set the default sorting order

[G2.2] Allow a user to globally activate or deactivate travel means.

[G2.3] Allow a user to intelligently rank the suggested solution.

[G3] Allow the users to define free time slot. (The user could set a specific time slot for lunch or personal affairs during a preset time slot).

[G4] Show the users the time which is going to be needed from one place to the next appointment or place, based on the users customized traffic preferences, the current traffic state and weather.

[G5] Push notifications to the users in time to make sure they acknowledge the coming appointments and reach the destination in time.

[G6] Allow the user to acquire the correct information about locations of bike sharing points, entrances of subways or bus station and so on.

[G6.1] Allow a user to sort the travel means based on time cost, economic cost and so on.

[G6.2] Allow a user to select combinations of transportation means.

[G6.3] Allow a user to select a travel mean and see the detail.

[G7] Allow a user to activate cloud service and share meetings on different devices.

[G7.1] Allow a user to add different devices to be shared.

[G7.2] Allow a user to set different permission on different devices, including create, read, write and delete.

[G8] Allow a user to active Big Data Analysis Service to retrieve better services from the system.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

User: People who is the client of this service. Before he/she start to use this system, he/she should insert some necessary information to help he/she get start:

-Preference on the solution of traffic

-Interests on the solution of traffic (cheapest/less walk/least carbon footprint/private car available)

-Reserved time slots

External System: Information system outside of TVLD but collaborate with TVLD to offer service. Including the weather information provider, difference traffic solution provider.

Reserved Time Slot: to optimize the user experience in TVLD the users are provided the ability to define a reserved time slot based on their own interest and considerations

Bike Sharing Point: The bike sharing point offered by bike renting services according to the location of the user (BikeMi. MoBike, ToBike…)

Meeting: An appointment with specific location and allotted time.

Effective Meeting: A meeting which is unreachable in the allotted time.

Ineffective Meeting: A meeting which is reachable in the allotted time.

Flexible Meeting: A meeting which is flexible in a range of allotted time.

Cloud Service: A service that allows user to share meetings on different devices at the same time.

Big Data Analysis Service: A cloud service that analyzes users' daily meetings and provides better supports for travel means suggestion.

1.3.2 Acronyms

TVLD: Travelender+, the target software we are designing in the RASD.

FTS: Free Time Slot

RASD: Requirement Analysis and Specification Document.

API: Application Programming Interface

1.3.3 Abbreviations

[Gn]: n-goal.

[Dn]: n-domain assumption.

[Rn]: n-functional requirement.

[Fn]: n-function

1.4 Revision History

TVLD is a new service in the nowadays software market. A great variety of schedule APPs could be found, nevertheless none of them offer a good solution for the integration of the information and resource from the external systems. But they could also provide a good reference in the design of TVLD.
Examples (System As-Is): Reminders (from Apple Inc.), Google Map (from Google Inc), …

1.5 Reference Documents

Teaching material from the Software Engineering 2 Professor Di Nitto
Specification Document: “Mandatory Project Assignments 2017 - 2018.pdf”.
GPS Performances: “http://www.gps.gov/systems/gps/performance/accuracy/".
UML Guidance: “http://www.html.it/guide/guida-uml/”.
StarUML Official Document: “http://docs.staruml.io/en/latest/”
Alloy Dynamic Model example: “http://homepage.cs.uiowa.edu/~tinelli/classes/181/Spring10/Notes/09-dynamic-models.pdf"
IEEE Std 830-1993 - IEEE Guide to Software Requirements Specifications.
IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications.

1.6 Overview

This RASD is composed by 6 parts and an appendix:
1. In the first part an introduction to the problem is given listing all the identified goals and providing some base information in order to better understand the other sections of the document.
2. The second part consists of an overall description of the system in which its boundaries are identified, and the actors involved in the system’s usage lifecycle are listed. The boundaries are given providing all the necessary assumptions: both the ones required in order to better understand the customer’s specifications given and the ones that will hold into the system and now on considered as true.
3. The third part is composed by the specific requirements identified, both functional and not functional.
4. In the fourth part a list of eight scenarios is provided; each of them describes a situation with the system might have to cope with.
5. The fifth part is entirely composed by the UML diagrams that model the system in detail.
6. Sixth part is embodied with the Alloy model of the system and includes all the relevant details; a proof of consistency and an example of the generated world are also provided.
7. The last part is accessory and contains a list of the tools used to redact this document and its contents, and a detailed report of the hours spent to do so.

2. Overall Description

2.1 Product Perspective

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

The system is developed dependently from the existed information systems. Generously, this system is a platform for the integration of information, developed with ODOO and external Map System (e.g. Google Map). The execution logic in TVLD should be simple and efficient. And the GUI should be designed friendly and easy to get start.

Users should insert correct and precise information into TVLD. TVLD should be able to handle the information along with the current information related to the users' requirements and calculate an optimal solution and remind the user in time, with different requirements of traffic reference or personal time slot taking into consideration.

2.2 Product functions

here we include the most important requirements

To make a specific boundary for TVLD, here in this part we offer a general view of the functions of our system. All goals should be fulfilled with the constrains under the functions provided by TVLD. As mentioned above, all goals we set in the first part are the major functions the system should offer. Here we focus on details of some ambiguous goals.

[F1] Store of personal information and settings of users
TVLD should be able to provide a variety of choices for the users to customize their trips. The information should be stored on their cellphone locally or in the server of TVLD. Both method should make sure that the information would be protected carefully and would not leased to other people. And the users should be able to modify and change their setting after the "get start" process. Besides the common appointments, TVLD offers the function of define personal free time slot.

[F2] Obtain and Integration information from external systems
TVLD should be able to obtain information from external system. The information should include (but not limited to) Weather Information, Traffic State, Location of Bike Sharing Points and Number of Bikes Available, Entrance of Subway, Bus Station.

[F3] Best Routine Selection
TVLD should be able to calculate the best routine from the users' current location. This involves the request for the users' cell phone should possess the GPS and 3G (or better, LTE, 4G and so on) functions so that TVLD could obtain the real-time information from external systems. And with all the online information necessary and

the customized preference settings, TVLD should always be able to calculate the best routine.

[F4] Notifications
TVLD should send notifications in the following situation

1. The user should start to move the next destination to reach the appointment in time with the routine offered by TVLD.
2. The users' predefined personal free time slot begins, the user could start to use this time to finish his /her own affairs (lunches, tea-break, …)
3. The user should get off the bus/get off the subway/return the bicycle and switch to another traffic method
4. The user has reached the destination

[F5] Big Data Analysis Service (OPTION FUNCTION)
Big Data Analysis Service provides a better service for users. If the service is active, the system will collect daily information from the user (including meeting information, chosen travel meals, user's preferences) and use Big Data to perform analysis. This will help to give more suitable suggested travel means according to user's behavior.

2.3 User characteristics

here we include anything that is relevant to clarify their needs

Users without initialization: people who have just download this software and haven't insert the necessary information, which means that TVLD could not offer intelligent service and give intelligent advices to them. TVLD can only work as a common reminder.

User: the people who have finished successfully the initialization steps and are access to all the service and functionalities offered by TVLD.

System Manager: the people who are in charge of the common affairs and business collaborations. Registration of this kind of users are done in the process of system installation.

External System (traffic method): share their states of their transportations.

External System (weather information source): share the states of the weather according to the location of user to support the choice of best routine.

2.4 Assumptions, dependencies and constraints

here we include domain assumptions

Only the functions (requirements) could not imply the desired goals. Here are the list of all the assumptions and constraints to help me this system have a clear view on its functions and execution logic.

2.4.1 Domain Assumptions

[D1] The location and time of the appointments set by the users are always correct and without ambiguity.
[D2] The traffic state and weather are predictable and would not change drastically in a relatively short time.
[D3] The information received in the TVLD are always correct and correctly reflect the current situation
[D4] TVLD could always tracking the traffic routes which it is planning and update the information continuously.
[D5] The algorithm implemented in TVLD is correct, always leading to the best solution.
[D6] The algorithm would not underestimate the time on the traffic.
[D7] When the time or location are modified, TVLD adapt based on the rest part of the schedules.
[D8] When the free time slot preset by the user are conflict with the appointment and could not be evaded by adapting another better solution, TVLD push a warning notification to the user.
[D9] Username must be unique.
[D10] The location and allotted time is assumed to be valid
[D11] A meeting is unreachable once the system cannot find any possible travel means to arrive in time.
[D12] The list of travel means will never be empty unless it is an unreachable meeting.
[D13] The basic travel means provided by external map system is always correct.
[D14] The meeting can only be ended once the user reaches the location.
[D15] All information collected from the user should be surely safe and anonymous.

2.4.2 Hardware Dependencies

Mobile App
- iOS or Android smartphone
- 2G/3G/4G connection
- GPS

Web App
- Modern browser able to retrieve user's location

3. Specific Requirements

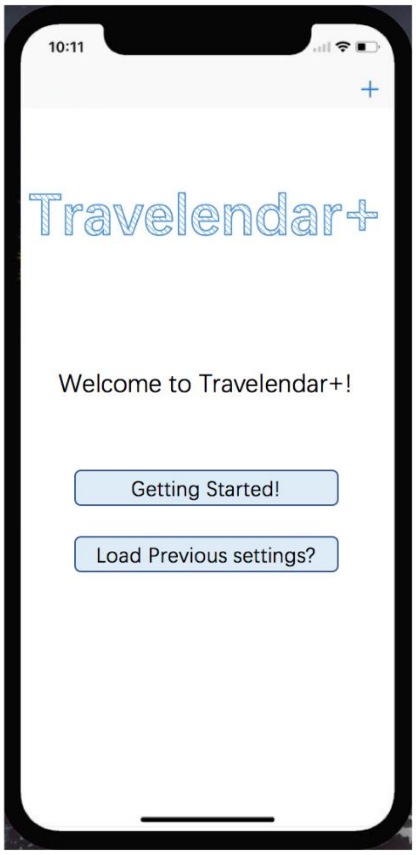
Here we include more details on all aspects in Section 2 if they can be useful for the development team

3.1 External Interface Requirements

3.1.1 User Interfaces

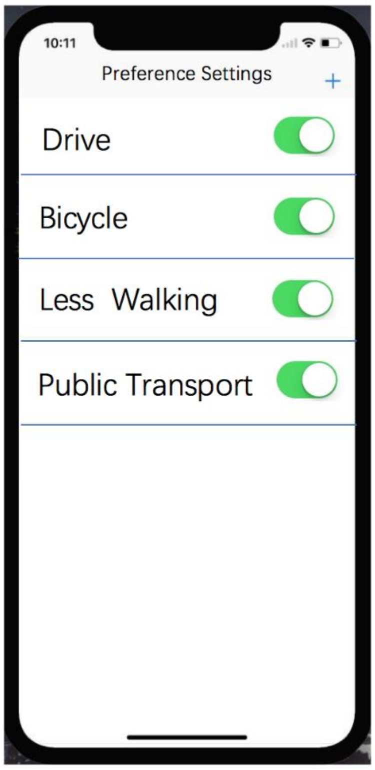
These following images demonstrate the users interface when the system first get released.

[UI1] User Login



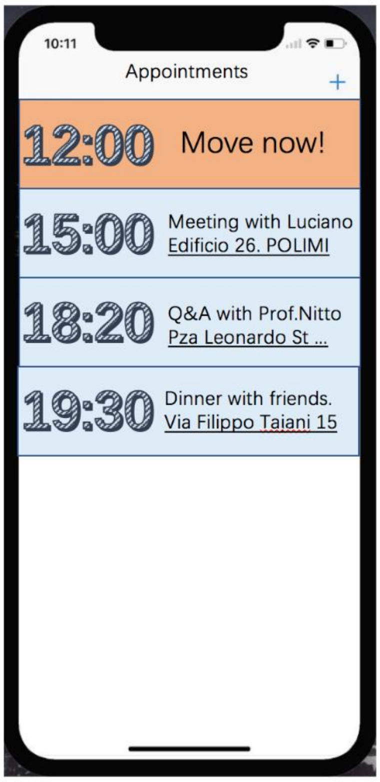
Give the user two options. “Getting Started” leads the user to the user’s preference settings. It helps the beginner the characteristic of TVLD. “Load Previous Settings” help the user to login and load their preference, inherit their appointments made on other devices.

[UI2] User Preference Settings



Here the user could define his/her own preference on the traffic methods. The user could disable some choices or select some choice to make the recommended route closer to the users' interest.

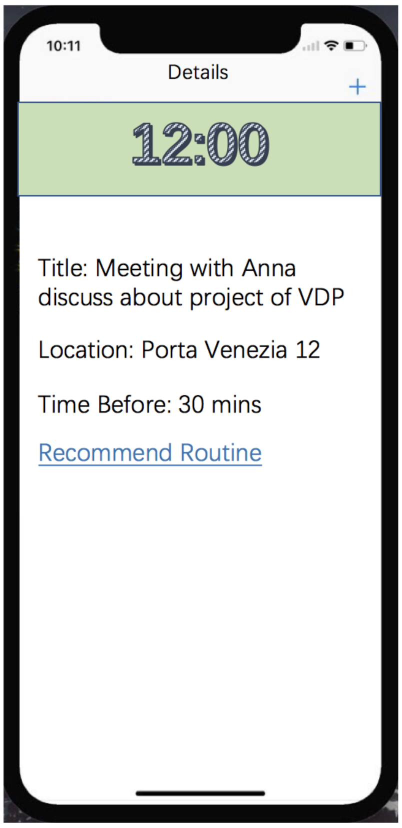
[UI3] Appointment Lists



12:00 Move now!
15:00 Meeting with Luciano
Edificio 26. POLIMI
18:20 Q&A with Prof.Nitto
Pza Leonardo St ...
19:30 Dinner with friends.
Via Filippo Tajani 15

Appointments are in blue blocks generously. If an event is coming soon and the user should get ready to move to the next location, the block would turn orange and shows information to urge the user to move ASAP.

[UI4] Appointment Details



12:00

Title: Meeting with Anna discuss about project of VDP

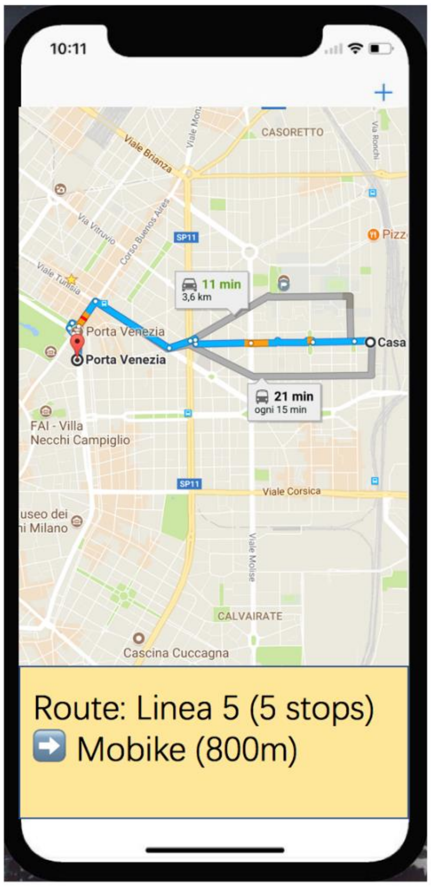
Location: Porta Venezia 12

Time Before: 30 mins

Recommend Routine

Here when the user clicked into one event, he could view the complete information about the appointments.

[UI5] Optimal Route



Highly integrated the information from other information resources and guide the user to the location accurately and efficiently.

3.1.2 Hardware Interfaces

The hardware is of great significance in this project, since we need the information of location, weather and some other things. While these interfaces could be replaced by the solutions on software interfaces.

3.1.3 Software Interfaces

GpsGate is one of the service provider on the GPS. It offers a cheap and feasible solution for the APP developer. They implemented scripting in the platform that lets you to do just that! This enables you to extend the functionality of GpsGate Server and make changes at any time. Or we can try to obtain the GPS information from the software on iPhone or Android which has already been implemented.

3.2 Functional Requirements

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

[G1] Allow users to set their time schedule including the location and time.

[R1] The user should be able to insert an appointment into his/her schedule

[R2] The user should be able to search an appointment which has been created.

[R3] The user should be able to cancel an appointment.

[R4] The user should be able to check information of appointments.

[G2] Allow the users to define their preference on the traffic method.

[R5] The user should be leaded to this page when they start to use this APP.

[R6] The user should be able to disable certain choices based on private situation.

[R7] The user should be able to make preference on the traffic methods.

[R8] All settings could be modified later after the initialization.

[G3] Allow the users to define free time slot. (The user could set a specific time slot for lunch or personal affairs during a preset time slot).

[R9] The user should be allowed to make a reminder of daily life by Monday/Tuesday/…

[R10] The user should be allowed to define a private time slot between a specific time.

[R11] The predefined events should always be take into consideration when TVLD giving advice.

[G4] Show the users the time which is going to be needed from one place to the next appointment or place, based on the users customized traffic preferences, the current traffic state and weather.

[R12] The user should be able to click into one appointment and check the details of the appointment including

-how much time it is going to take to travel there

-what is the recommended routine from here to the place

-how much time before travel there

[R13] The users should be allowed to check the routine time in detail like the travelling method and the reason why it is going to take such time (weather, traffic strikes…)

[G5] Push notifications to the users in time to make sure they acknowledge the coming appointments and reach the destination in time.

[R14] The user should be notified when TVLD is offline and could not offer and intelligent services.

[R15] The user should be notified when it is time to move to next destination.

[R16] The user should be notified when the user arrived the destination.

[G6] Allow the user to acquire the correct information about locations of bike sharing points, entrances of subways or bus station and so on.

[R17] The user should access to the accurate location of the traffic methods on the map with the help of TVLD.

[R18] The user should be notified when it is time to switch a traffic method

[G7] Allow a user to activate cloud service and share meetings on different devices.
[R19] The user should be allowed to update to cloud services and make all the devices with same account access to the service without change setting and preferences again.

[G8] Allow a user to active Big Data Analysis Service to retrieve better services from the system.
[R20] The user should be able to allow the TVLD developer groups use the data of the user and generate methods to improve the service quality.

3.3 Performance Requirements

The system offer service facing the very single users. Most of the calculation of the service would be executed on the mobile devices of the user. Since this system is designed to help people manage their schedule and avoid being late, the service provided should be 24/7 (over 5-nines) and the basic functions should be provided even without network. On the same time, during the use of this information system, the system should be allowed to use the location information of the users, which means that our system should be very careful to preserve the private information and will not use it for another commercial use.

3.4 Design Constraints

3.4.1 Standards Compliance

The system is designed in the environment Python and Java. While the implementation of this system may require the ability in Swift or Object C. Generally, the system should fulfill all the release agreement on the market of different software like Google Play or Apple Store.

3.4.2 Any Other Constraints

Some of the external information are open source. This information would be available on their website totally free (for example, the Google Map and the weather information). However, some information is very confidential in the owner of the companies (for example, the MoBike company). Which means we may have to alternate our strategy in the design of this system to collect the information and generate the optimal travel solutions.

3.5 Software System Attributes

3.5.1 Reliability

The system would be released in the Google Play and Apple store. Users of interests should have an easy access to the system. And here the TVLD should offer a service in a really great range and should be able to performance some basic functions even without network. At least it should notify the users information like “Hey you get no network, I cannot help you to plan your routine to your destination, but you got only 1 hour left!”. With the network, TVLD should be able to execute the required functions and the suggestions it provide should be of great accuracy.

3.5.2 Availability

The system must guarantee a 24/7 service. Very small deviations from this requirement will be obviously acceptable.

3.5.3 Security

Users credentials and traffic preference will be stored. Security of the data and of the routine of traffic is a primary concern. Or there may be a potential danger that some people would track certain user by this App.

3.5.4 Maintainability

The system should be easy to maintain. But TVLD has a priority in its reliability and availability, so the maintain should be easy to execute and the whole system should be online at the very same time of maintenances.

3.5.5 Portability

TVLD is designed to be implemented on mobile devices, it is very promising to attract users with need to better arrange their daily routines and guaranteed an integrated level of information to choose while travelling.

4. Scenarios

4.1 Scenario 1

Alice is a student of Polytechnic of Milan. She lives in the apartment 5 blocks away from POLIMI. Her friend Bob advise her to use TVLD. This morning she download one with her smartphone. TVLD notified Alice to register to have a better performance on TVLD. During the registration process, Alice was asked her preference on traffic, and disable the DRIVE choice because Alice told TVLD that she had not gotten a driving license. And she also mentioned that she would prefer a routine with less walking because she did not like to walk. After these, Alice inserted that she had a course today in Edificio 1 of POLIMI. TVLD pushed her a notification after Alice finished her breakfast and she was suggested to take a subway. She followed the guidance of TVLD and finally Alice arrive the classroom in time.

4.2 Scenario 2

Caroline is a registered user of TVLD. She is a secretary of Bank. She preset a 1-hour free time slot from 12:00 to 14:00 to take a short break and have lunch. While today, she was notified to go to the post office and retrieve a package from 13:00 to 14:00. So, she inserted this in her appointment today. At the beginning of her lunch break, Caroline was planning to finish all the works and then go to lunch. While she got notified from her TVLD and suggests her that she should take a lunch now because she was going to retrieve a package later. So, Caroline follow the suggestions showed and did not miss a thing.

4.3 Scenario 3

Dario got up at 7 o’clock as usual. While he was taking the breakfast, he got a notified from TVLD shows that he should prepare to move to his work now, because there is a strike on the subway, so the time driving from Dario’s apartment to his work is going to be longer than usual. Dario finished his breakfast quickly and left his apartment 15 minutes earlier than usual. And Dario arrived his workplace in time.

4.4 Scenario 4

Elsa is a teaching assistant from POLIMI. She was told by the professor to come to the professor’s office at 14:00 and she inserted the appointment quickly. So Elsa postpone her yoga course in the gym which was supposed to be 14:30. However at 12:00 the professor told Elsa that the meeting was cancelled. When Elsa checked it with professor and confirmed it, she cancelled the appointments at 14:00. At 13:45 she received the notification from TVLD suggesting that it was time for her to move to the gym by bus line 5. As a result, she reached gym in time.

4.5 Scenario 5

Federico set an appointment that he should return to the city center to a party at 19:00. However, he forgot to charge his telephone number this month. His network service was disabled, and he still have not notice that. TVLD sent Federico a notification that TVLD cannot offer him the online services at the offline state. But Federico was still notified by TVLD 2 hours in previous about the party, and he had enough time to prepare and pick clothes. Though without the assistance of TVLD on the trip routines, still Federico arrived the party in time.

4.6 Scenario 6

Garlleo started to use TVLD after one of his friends suggested him. But he forgot to customize his private TVLD settings, and he did not want to do it without WIFI. So Garlleo decided to skip the private settings, use the basic functions of TVLD just like other reminders. TVLD perfectly finished these tasks and at night, Garlleo could start to customize his setting and started to use the advance functions.

4.7 Scenario 7

Henry has been using TVLD for over 3 months. In the "getting start pages" he activates the service that TVLD could collect information about his daily route, though at that time he did not understand what it meaned. While TVLD after generating enough information of Henry, it generated a report of Henry's daily life and when he opened TVLD, TVLD pushed Henry a notice that he always has lunch so irregularly. TVLD help Henry to rearrange his appointment and try to find a solution for Henry to have lunch regularly, and even tell Henry the address of the vegetarian restaurants nearby.

5. UML Modeling

5.1 Use Case Description

5.1.1 User creates appointment

ACTORS	User/Visitor
GOALS	[G1]
INPUT CONDITIONS	The user is in the Appointments page.
EVENTS FLOW	1. The user clicks the "+" button on the upper right corner of the Appointments page. 2. The system directs the user to the Appointments Creation page. 3. The user enters "Title", "Location" and for the appointment. 4. The user clicks the "Create" button. 5. The system creates an appointment in local database and redirects the user to the Appointments page.
OUTPUT CONDITIONS	An appointment is successfully created.
EXCEPTIONS	1. The location of the appointment cannot be reached in the allotted time. This will be handled by showing a warning message on the screen and asking the user to modify the information.

5.1.2 User defines preference

ACTORS	User
GOALS	[G2]
INPUT CONDITIONS	The user has logged in and is in the Appointments page.
EVENTS FLOW	1. The user clicks the "Setting" button on the upper left corner of the Appointments page. 2. The user can choose preference on their travel means. 3. The user can disable certain choices based on private situation. 4. The user clicks the "Save" button to save the setting.

	5. The system redirects the user to the Appointments page.
OUTPUT CONDITIONS	The preference setting is finally saved.
EXCEPTIONS	

5.1.3 User creates free time slot

ACTORS	User
GOALS	[G3]
INPUT CONDITIONS	The user has logged in and is in the Appointments page.
EVENT FLOW	1. The user clicks the “Setting” button on the upper left corner of the Appointments page. 2. The user clicks the “Create Free Time Slot” button. 3. The user enters “Title” and “Time Slot” and choose the activation days. 4. The user clicks the “Create” button. 5. The system creates a free time slot in local database and redirects the user to the Appointments page.
OUTPUT CONDITIONS	A free time slot is successfully created.
EXCEPTIONS	1. The free time slot has a time conflict with other free time slot. This will be handled by showing a warning message on the screen and asking the user to modify the information.

5.1.4 User creates appointment’s detail

ACTORS	USERS
GOAL	[G4]
INPUT CONDITIONS	The user has created an appointment.
EVENT FLOW	1. The user clicks into one appointment. 2. The system directs the user to the “Appointment Detail” page. 3. The system shows the “Title”, “Location”, “Travel Time”, “Time Before” fields and a “Recommended Routine” button on the page.
OUTPUT CONDITIONS	The user has checked the details of the appointment.

EXCEPTIONS	

5.1.5 User receives notification of the appointment

ACTORS	User/Visitor
GOALS	[G5]
INPUT CONDITIONS	The user has created an appointment and the appointment's time is coming closely according to the travel time.
EVENT FLOW	1. The system pushes a message to the user about the coming appointment. 2. The system pushes a message to the user when the user reaches the destination.
OUPUT CONDITIONS	The user receives notification about the coming appointment from the system.
EXCEPTIONS	1. The user hasn't logged in and is considered as a visitor. 2. The TVLD system is offline. These will be handled by pushing a message to the user according to a constant time slot set by the user, but not according to the travel time.

5.1.6 User follows routine

ACTORS	User
GOALS	[G6]
INPUT CONDITIONS	The user has created an appointment and is in the "Appointment Detail"
EVENT FLOW	1. The user clicks the "Recommended Routine" button on the page. 2. The user clicks into one recommended routine from the routines list. 3. The system directs the user to the "Routine" page and displays the routine by a map. 4. The system locates the user on the map continuously. 5. The system pushes message to the user about switching the traffic method.

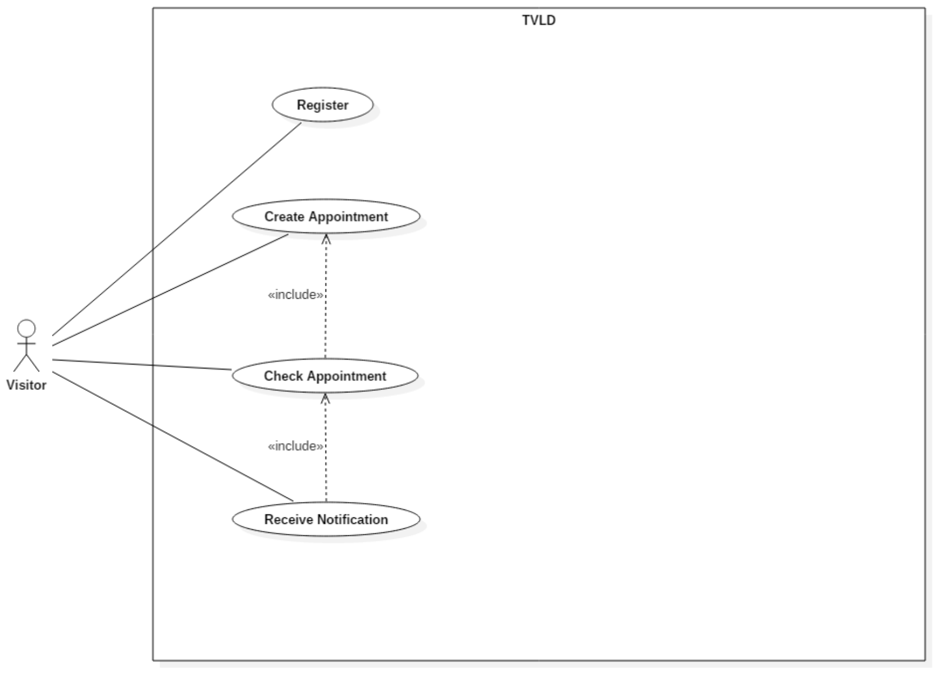
OUTPUT CONDITIONS	The user acquires the detail for the travel means.
EXCEPTIONS	1. The user's device doesn't turn GPS on. This will be handled by showing an error message for that and asking to user to turn GPS on.

5.1.7 User uses cloud service

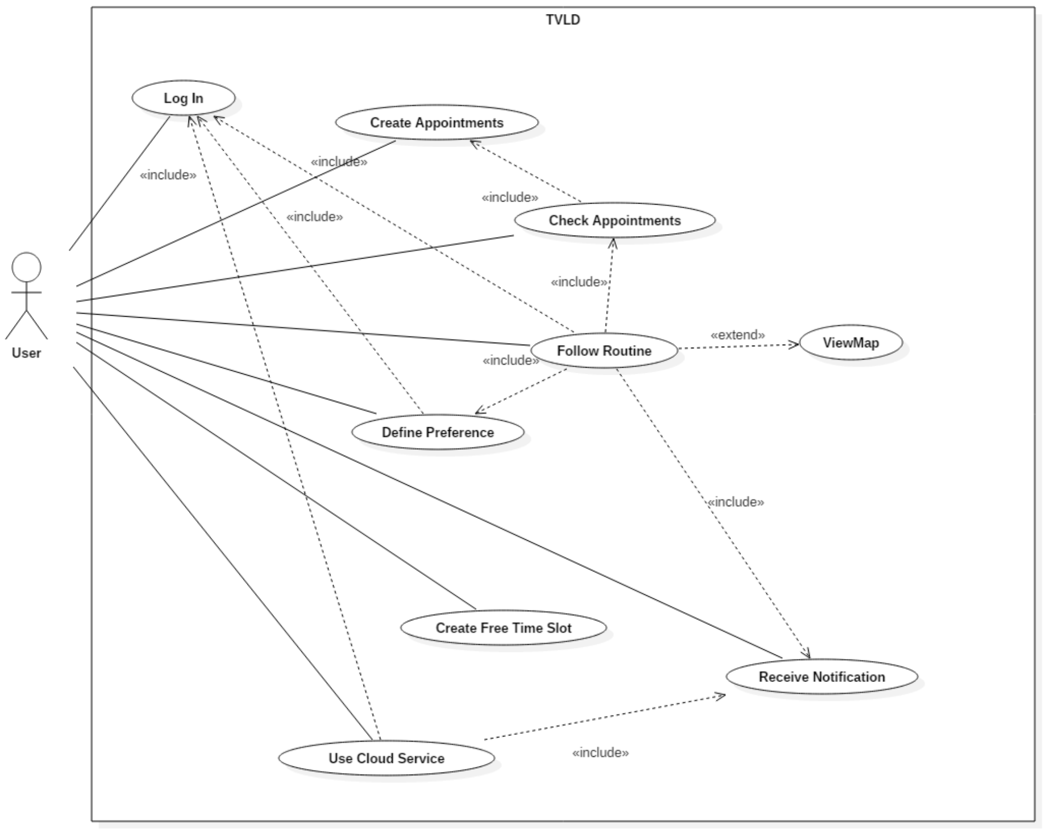
ACTORS	User
GOALS	[G7]
INPUT CONSTRAINTS	The user has logged in and activated the cloud service.
EVENT FLOW	1. The system will continuously upload user's data anonymously to the cloud server for analysis. 2. The system receives from the cloud server both travel suggestion and schedule suggestion based on user's habit. 3. The system pushes these suggestions to the user.
OUTPUT CONSTRAINTS	The user receives travel/schedule suggestion based on cloud service.
EXCEPTION	

5.2   Use Case Diagram

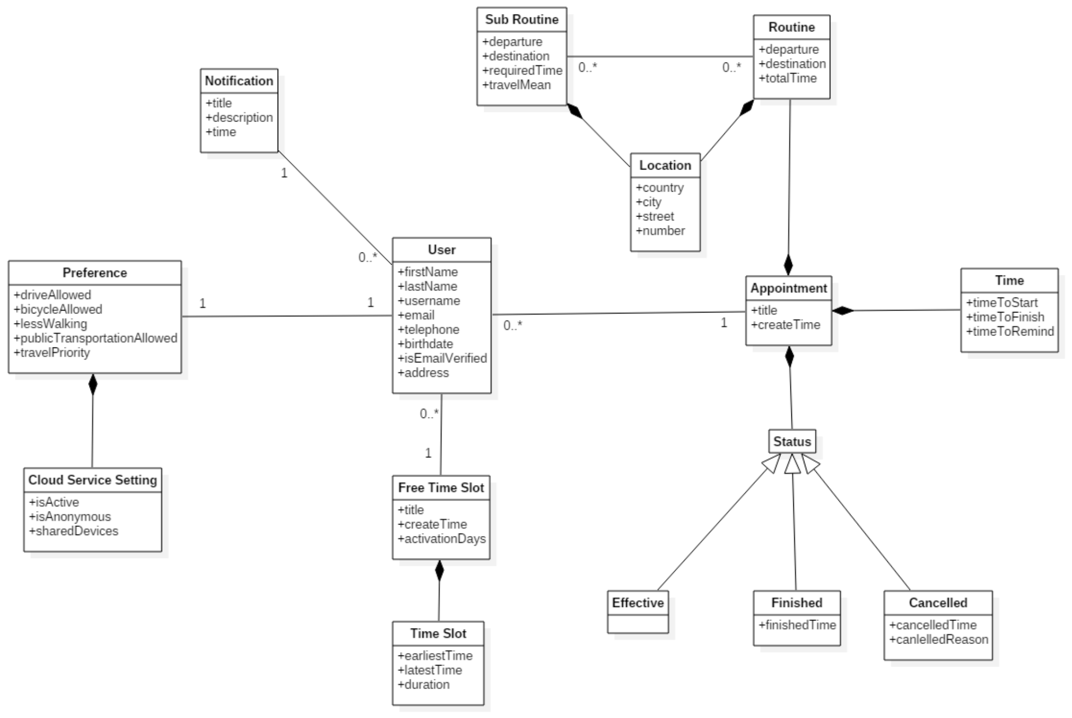
5.2.1 Visitor



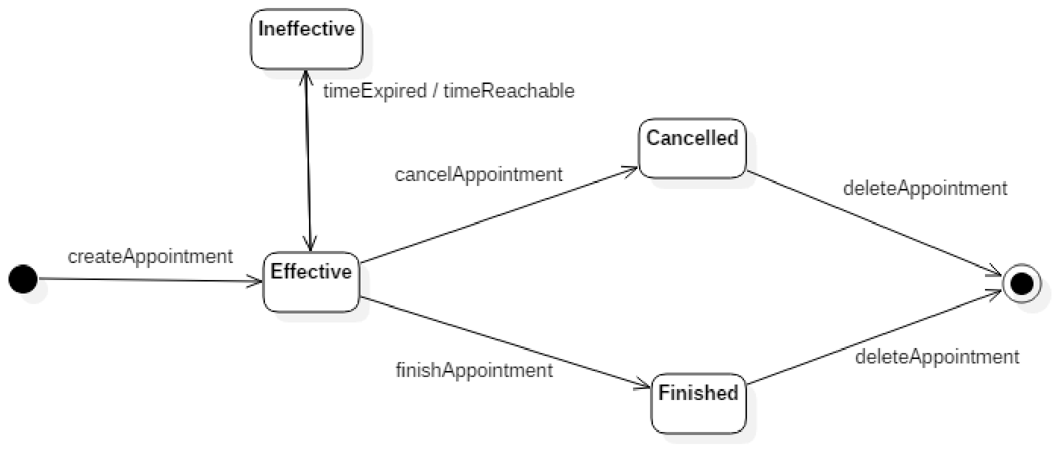
5.2.2 User



5.3 Class Diagram

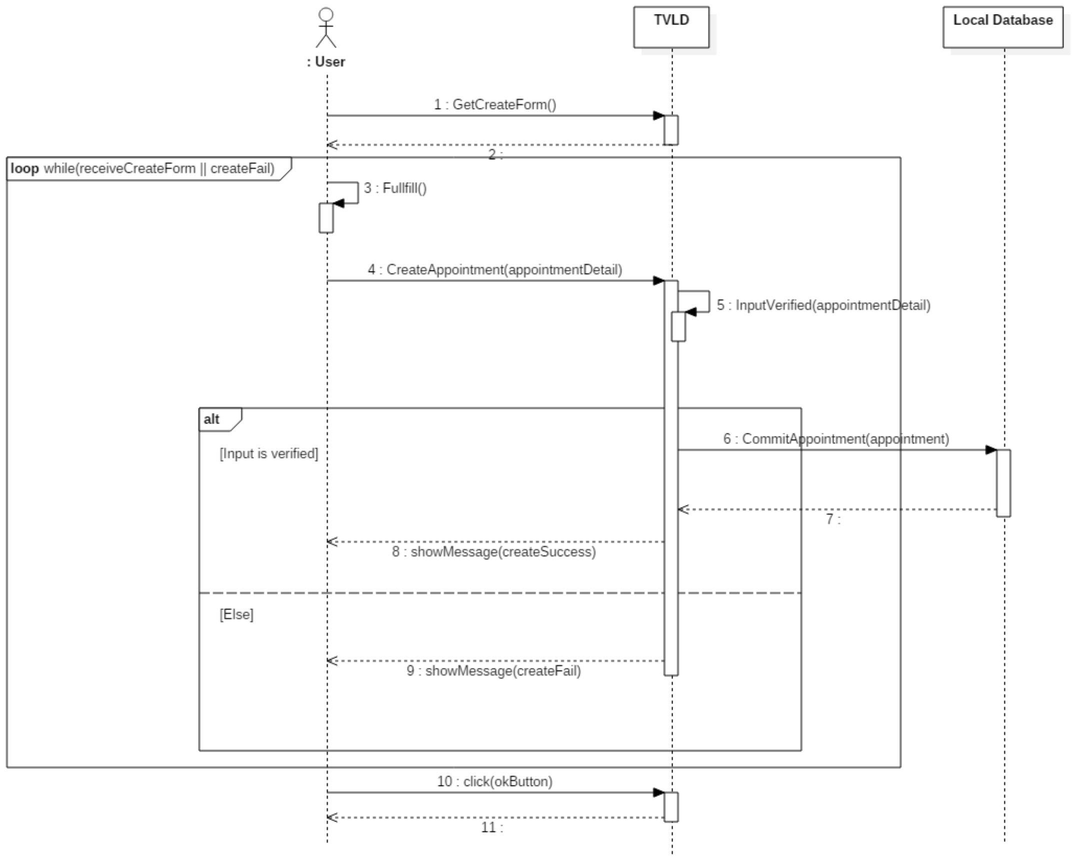


5.4 Statechart Diagram

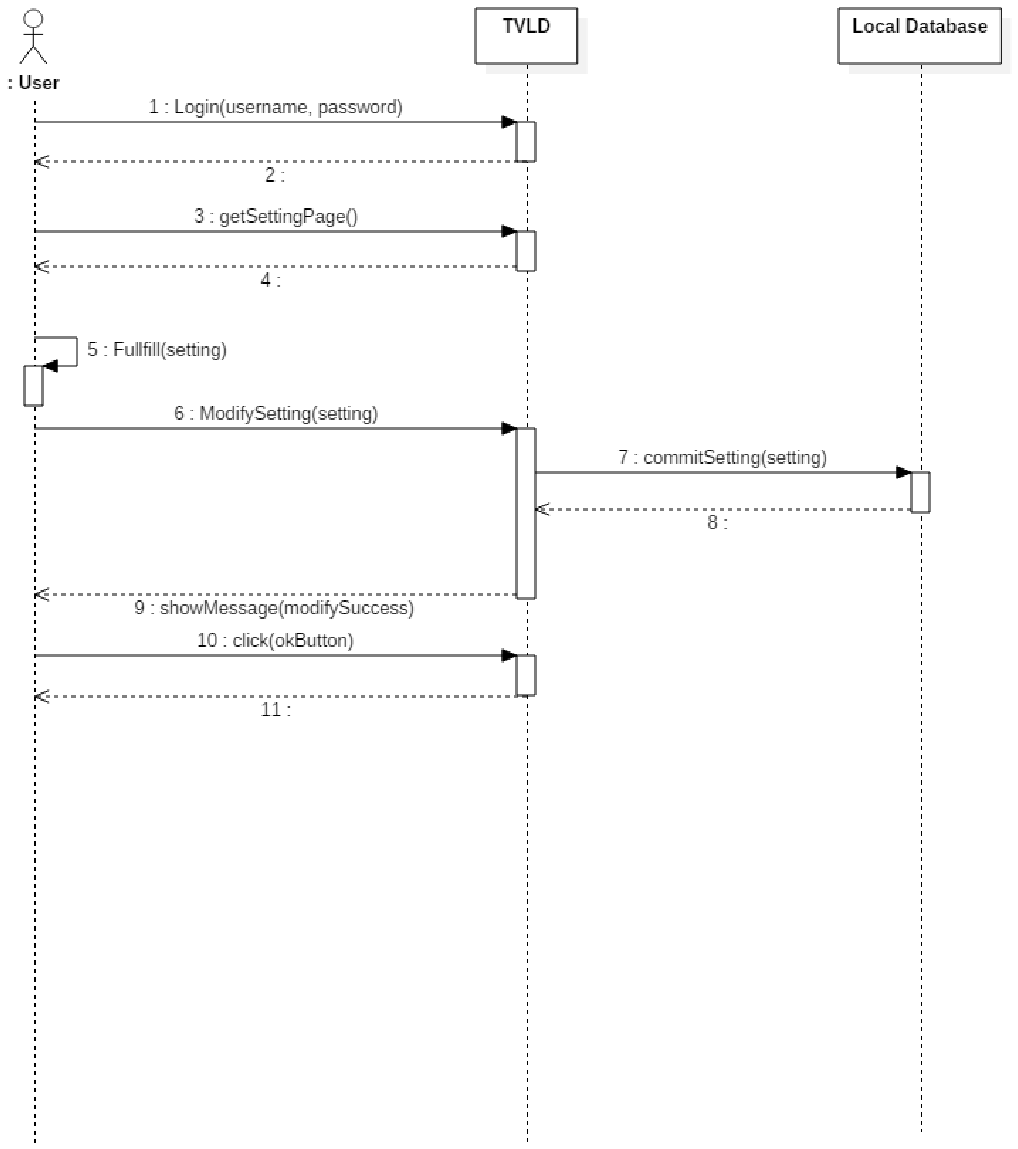


5.5 Sequence Diagram

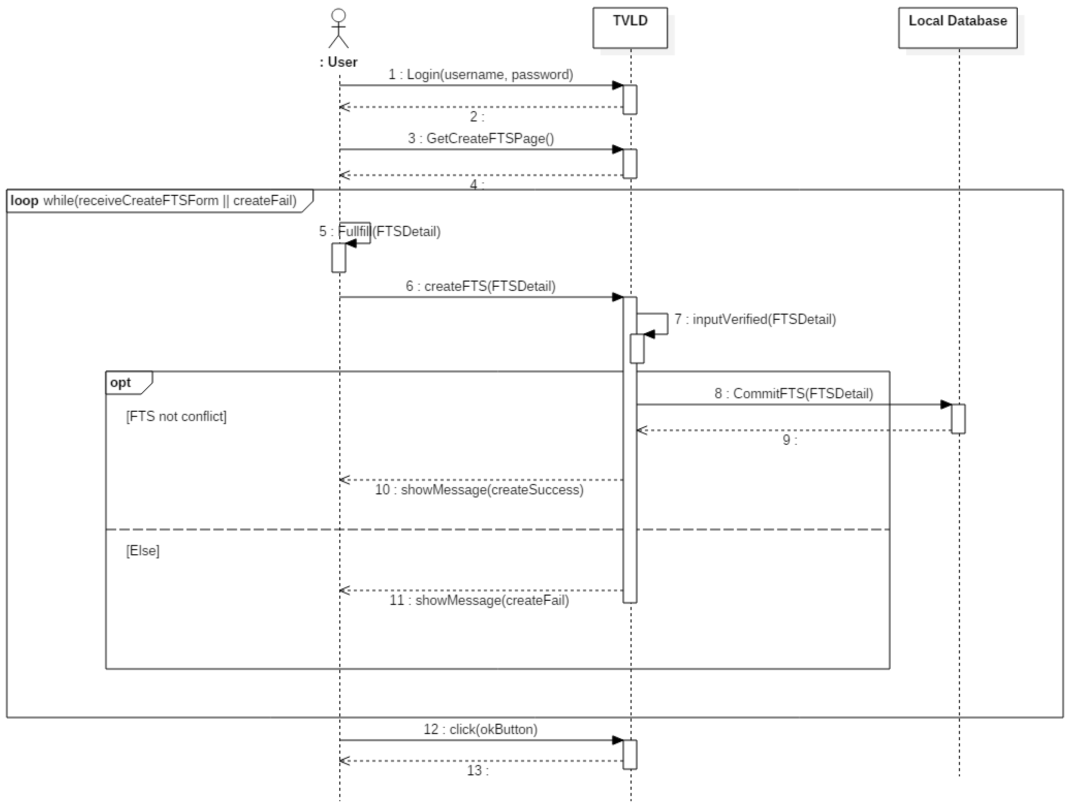
5.5.1 User creates appointment


: User
TVLD
Local Database
1 : GetCreateForm()
2 :
loop while(receiveCreateForm || createFail)
3 : Fullfill()
4 : CreateAppointment(appointmentDetail)
5 : InputVerified(appointmentDetail)
alt
[Input is verified]
6 : CommitAppointment(appointment)
7 :
8 : showMessage(createSuccess)
[Else]
9 : showMessage(createFail)
10 : click(okButton)
11 :

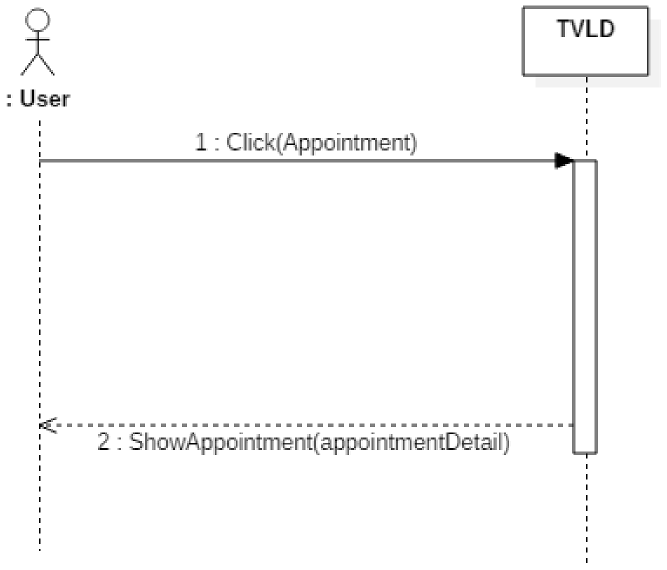
5.5.2 User defines preference


: User
TVLD
Local Database
1 : Login(username, password)
2 :
3 : getSettingPage()
4 :
5 : Fullfill(setting)
6 : ModifySetting(setting)
7 : commitSetting(setting)
8 :
9 : showMessage(modifySuccess)
10 : click(okButton)
11 :

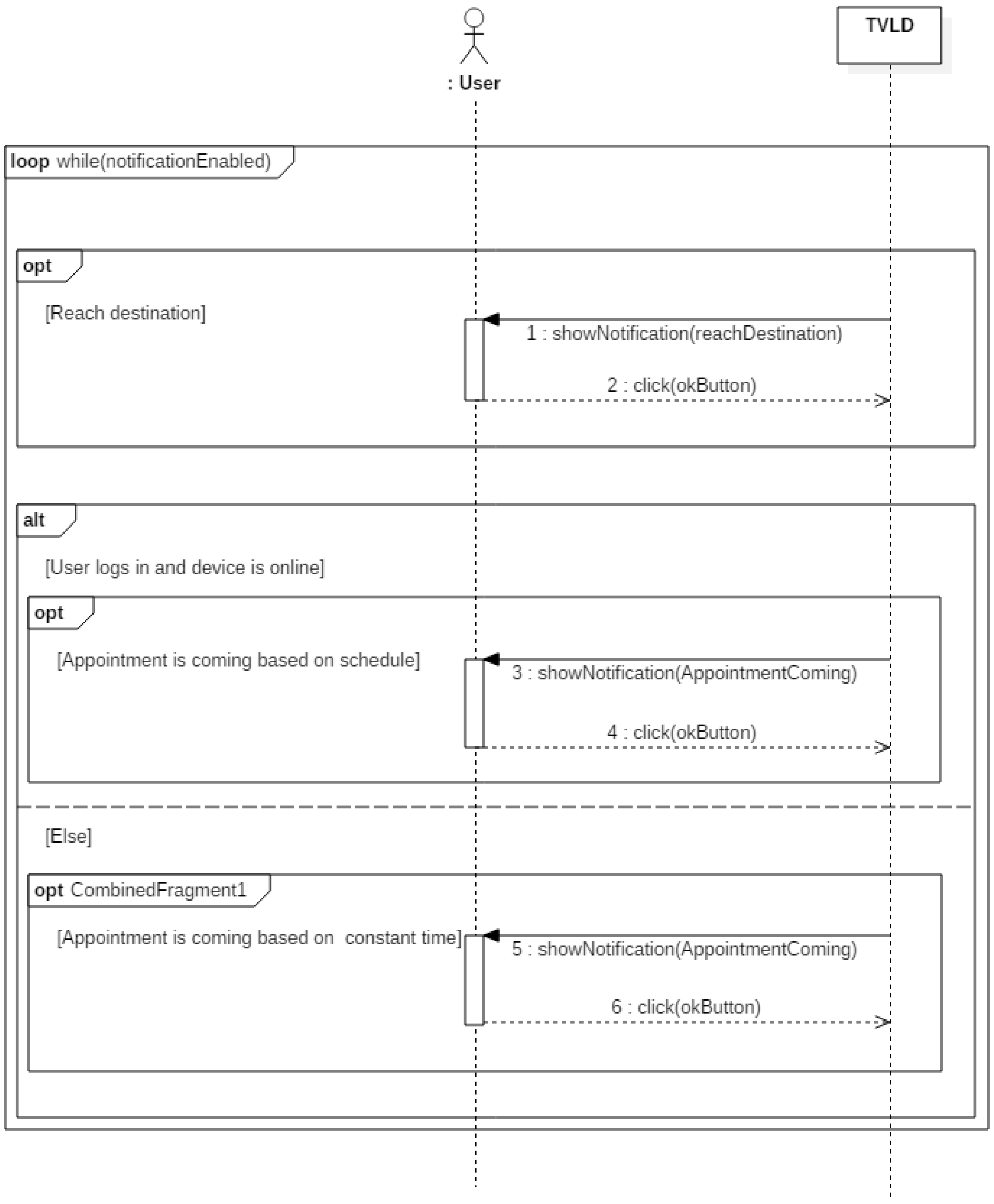
5.5.3 User creates free time slot



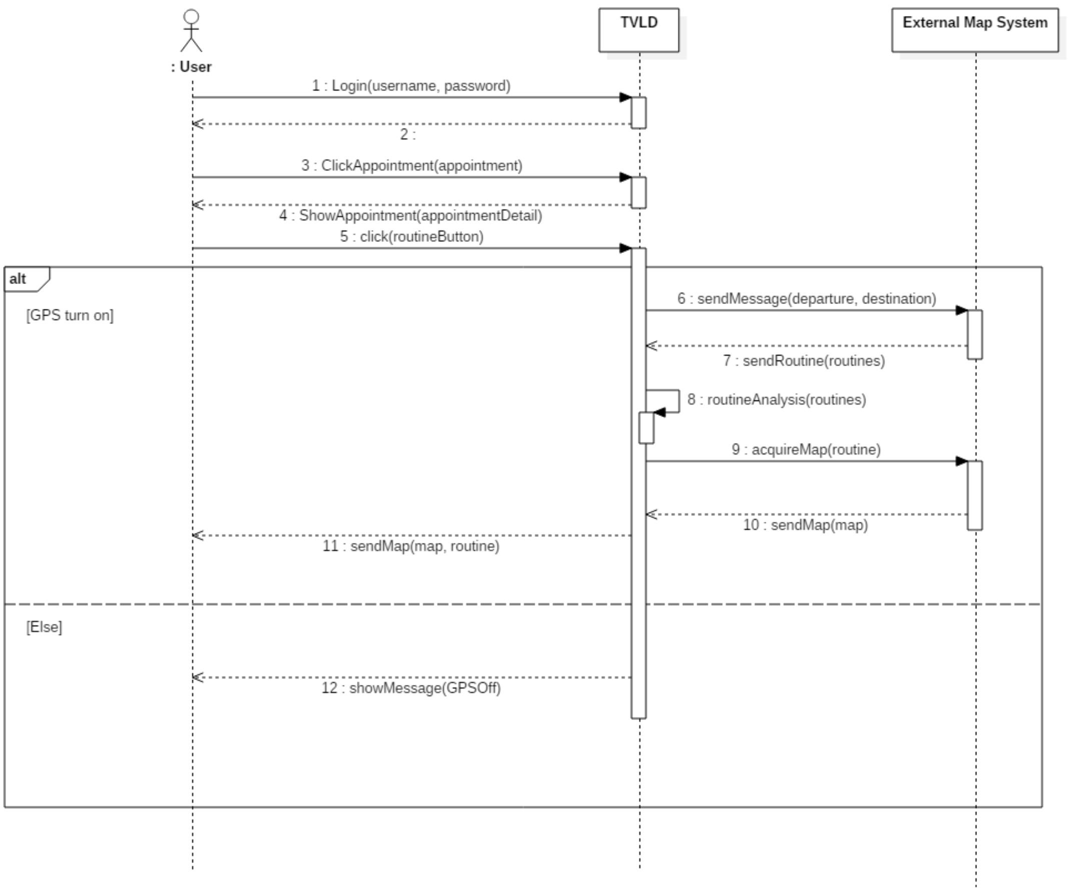
5.5.4 User checks appointment's detail



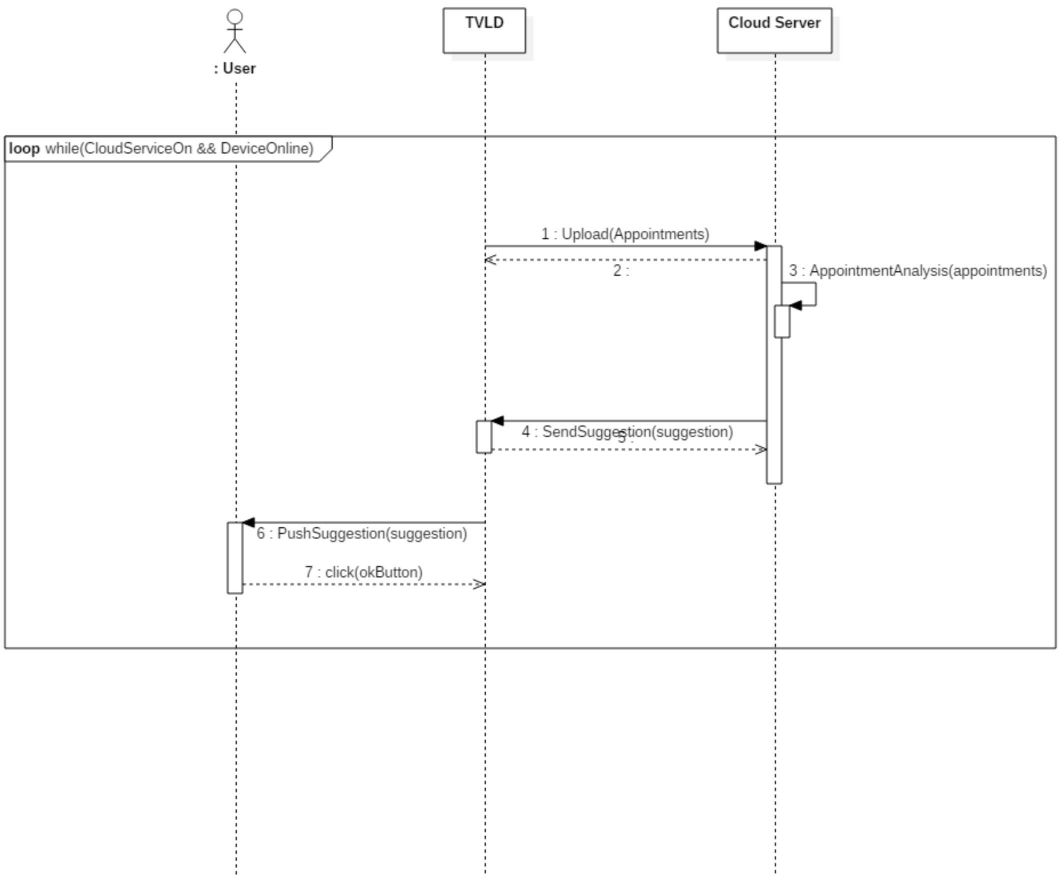
5.5.5 User receive notification of the appointment


: User
TVLD
loop while(notificationEnabled)
opt
[Reach destination]
1 : showNotification(reachDestination)
2 : click(okButton)
alt
[User logs in and device is online]
opt
[Appointment is coming based on schedule]
3 : showNotification(AppointmentComing)
4 : click(okButton)
[Else]
opt CombinedFragment1
[Appointment is coming based on constant time]
5 : showNotification(AppointmentComing)
6 : click(okButton)

5.5.6 User follows routine


: User
TVLD
External Map System
1 : Login(username, password)
2 :
3 : ClickAppointment(appointment)
4 : ShowAppointment(appointmentDetail)
5 : click(routineButton)
alt
[GPS turn on]
6 : sendMessage(departure, destination)
7 : sendRoutine(routines)
8 : routineAnalysis(routines)
9 : acquireMap(routine)
10 : sendMap(map)
11 : sendMap(map, routine)
[Else]
12 : showMessage(GPSOff)

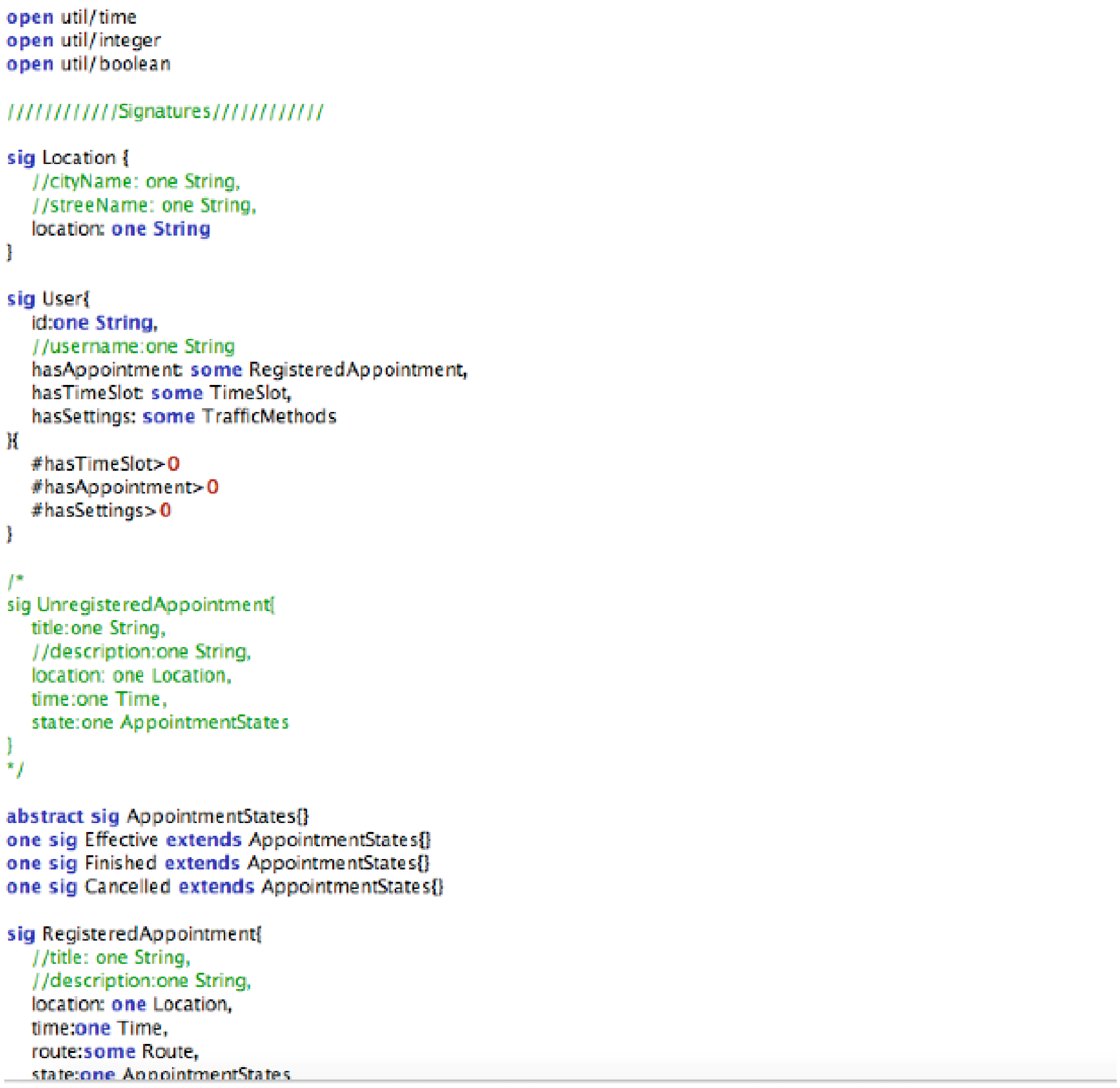
5.5.7 User uses cloud service



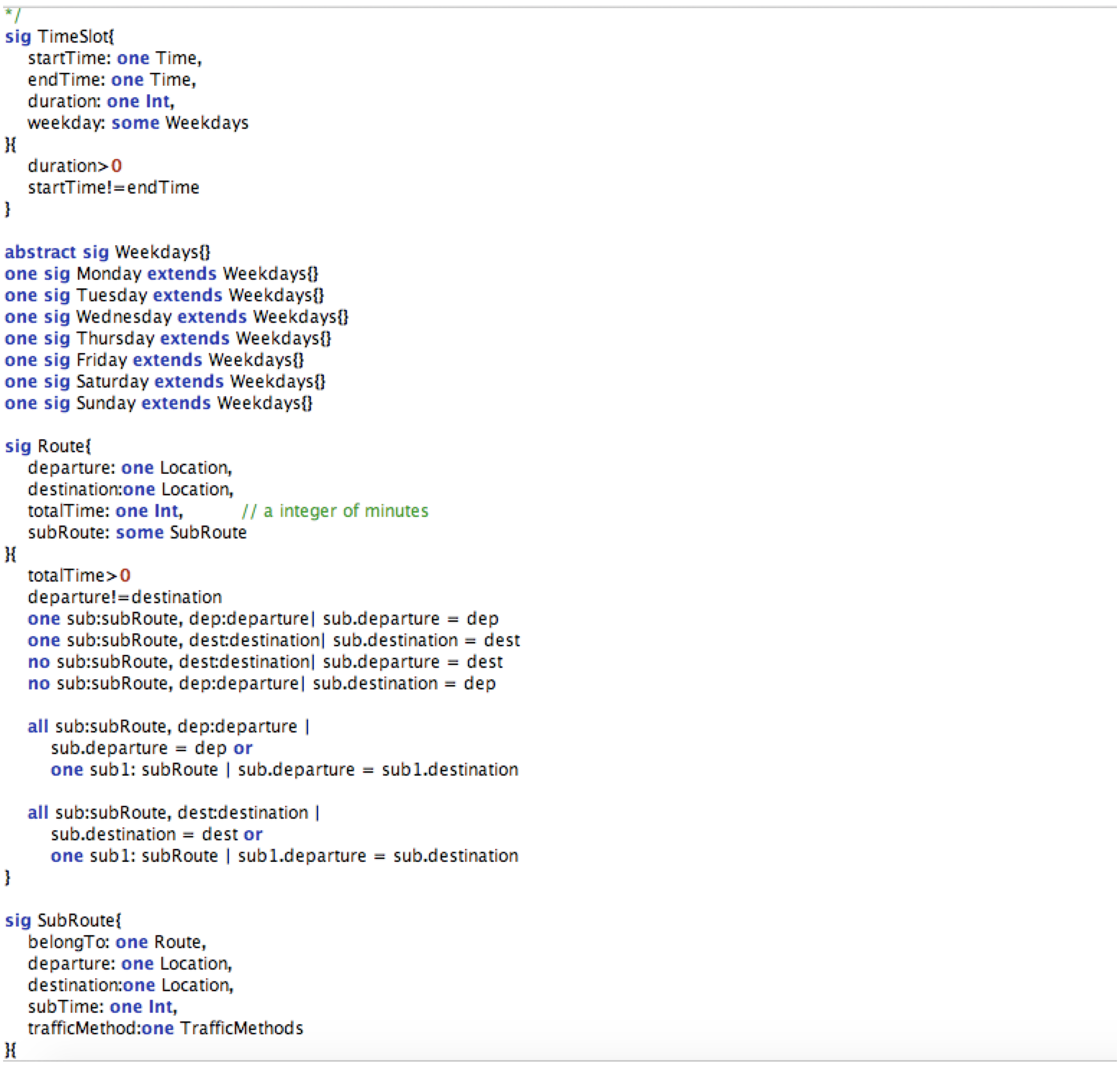
: User
TVLD
Cloud Server
loop while(CloudServiceOn && DeviceOnline)
1 : Upload(Appointments)
2 :
3 : AppointmentAnalysis(appointments)
4 : SendSuggestion(suggestion)
5 :
6 : PushSuggestion(suggestion)
7 : click(okButton)

6. Alloy Modeling

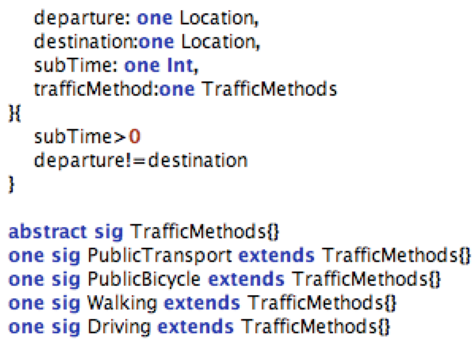
6.1 Signatures



```
open util/time
open util/integer
open util/boolean

////////////Signatures////////////

sig Location {
    //cityName: one String,
    //streeName: one String,
    location: one String
}

sig User{
    id:one String,
    //username:one String
    hasAppointment: some RegisteredAppointment,
    hasTimeSlot: some TimeSlot,
    hasSettings: some TrafficMethods
}{
    #hasTimeSlot>0
    #hasAppointment>0
    #hasSettings>0
}

/*
sig UnregisteredAppointment{
    title:one String,
    //description:one String,
    location: one Location,
    time:one Time,
    state:one AppointmentStates
}
*/

abstract sig AppointmentStates{}
one sig Effective extends AppointmentStates{}
one sig Finished extends AppointmentStates{}
one sig Cancelled extends AppointmentStates{}

sig RegisteredAppointment{
    //title: one String,
    //description:one String,
    location: one Location,
    time:one Time,
    route:some Route,
    state:one AppointmentStates
```

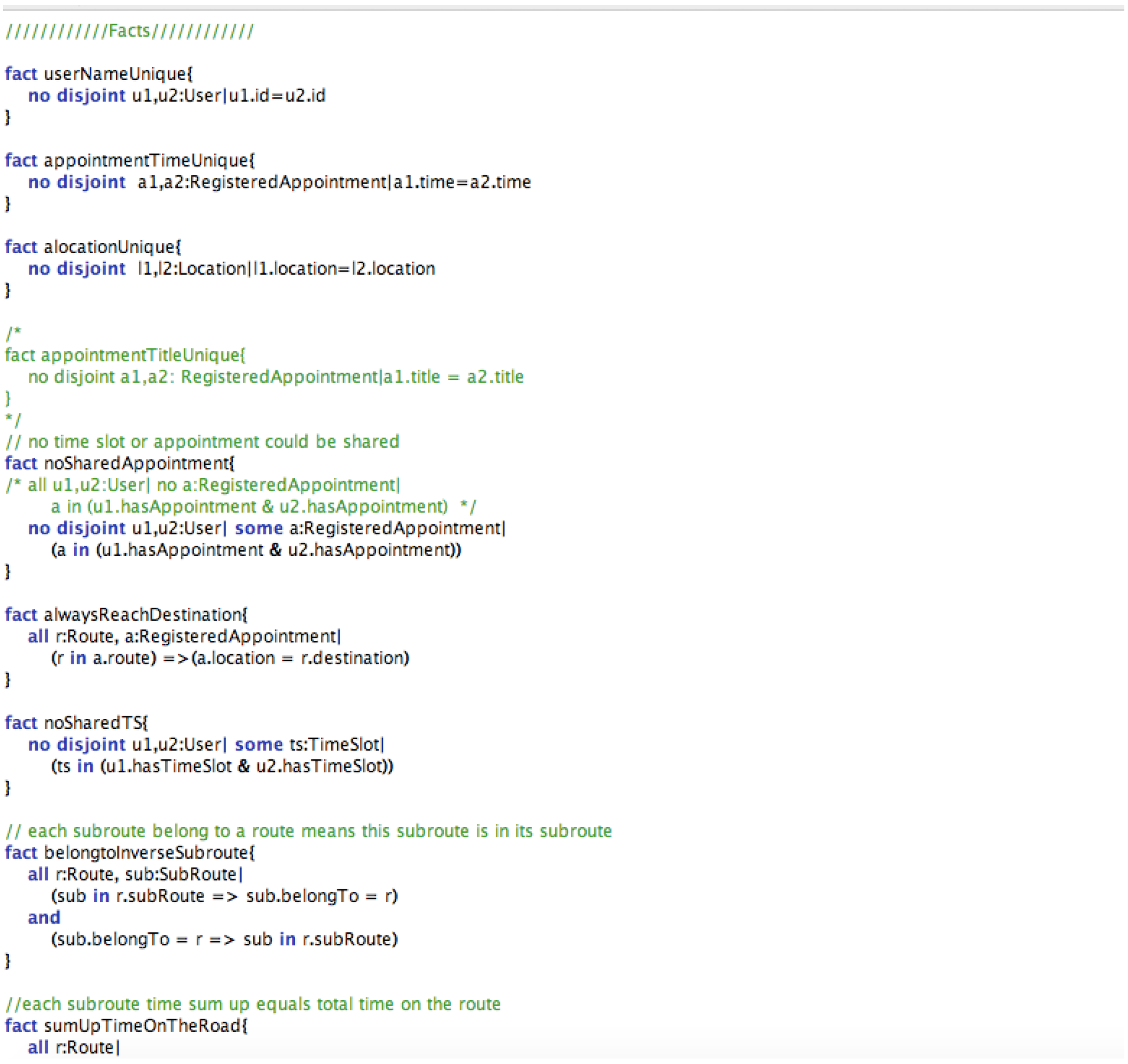


```
*/
sig TimeSlot{
    startTime: one Time,
    endTime: one Time,
    duration: one Int,
    weekday: some Weekdays
}{
    duration>0
    startTime!=endTime
}

abstract sig Weekdays{}
one sig Monday extends Weekdays{}
one sig Tuesday extends Weekdays{}
one sig Wednesday extends Weekdays{}
one sig Thursday extends Weekdays{}
one sig Friday extends Weekdays{}
one sig Saturday extends Weekdays{}
one sig Sunday extends Weekdays{}

sig Route{
    departure: one Location,
    destination:one Location,
    totalTime: one Int,          // a integer of minutes
    subRoute: some SubRoute
}{
    totalTime>0
    departure!=destination
    one sub:subRoute, dep:departure| sub.departure = dep
    one sub:subRoute, dest:destination| sub.destination = dest
    no sub:subRoute, dest:destination| sub.departure = dest
    no sub:subRoute, dep:departure| sub.destination = dep

    all sub:subRoute, dep:departure |
        sub.departure = dep or
        one sub1: subRoute | sub.departure = sub1.destination

    all sub:subRoute, dest:destination |
        sub.destination = dest or
        one sub1: subRoute | sub1.departure = sub.destination
}

sig SubRoute{
    belongTo: one Route,
    departure: one Location,
    destination:one Location,
    subTime: one Int,
    trafficMethod:one TrafficMethods
}{
```



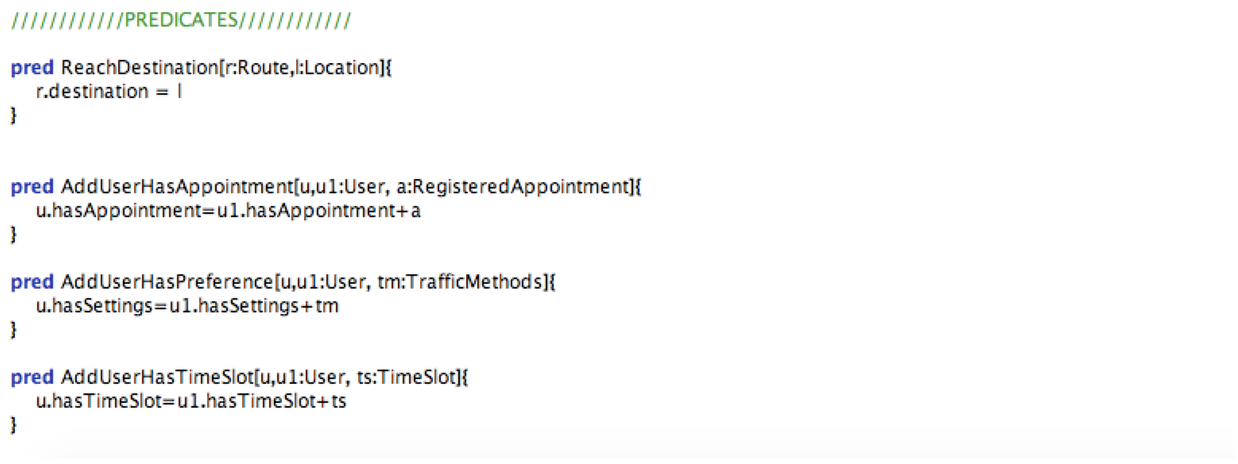
```
    departure: one Location,
    destination:one Location,
    subTime: one Int,
    trafficMethod:one TrafficMethods
}{
    subTime>0
    departure!=destination
}

abstract sig TrafficMethods{}
one sig PublicTransport extends TrafficMethods{}
one sig PublicBicycle extends TrafficMethods{}
one sig Walking extends TrafficMethods{}
one sig Driving extends TrafficMethods{}
```

6.2 Facts

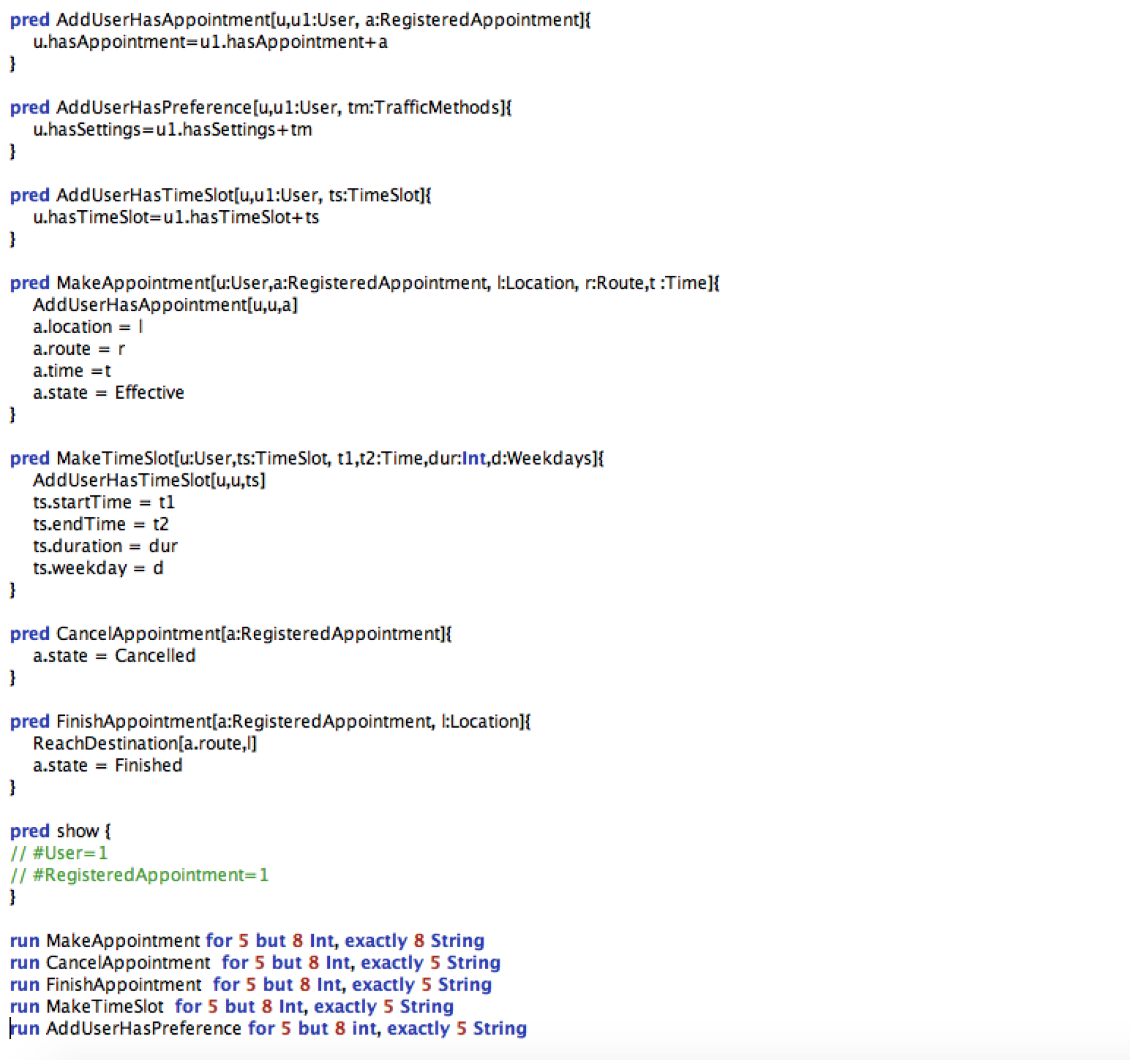


```
////////////Facts///////////

fact userNameUnique{
    no disjoint u1,u2:User|u1.id=u2.id
}

fact appointmentTimeUnique{
    no disjoint  a1,a2:RegisteredAppointment|a1.time=a2.time
}

fact alocationUnique{
    no disjoint  l1,l2:Location|l1.location=l2.location
}

/*
fact appointmentTitleUnique{
    no disjoint a1,a2: RegisteredAppointment|a1.title = a2.title
}
*/
// no time slot or appointment could be shared
fact noSharedAppointment{
/* all u1,u2:User| no a:RegisteredAppointment|
        a in (u1.hasAppointment & u2.hasAppointment)  */
    no disjoint u1,u2:User| some a:RegisteredAppointment|
        (a in (u1.hasAppointment & u2.hasAppointment))
}

fact alwaysReachDestination{
    all r:Route, a:RegisteredAppointment|
        (r in a.route) =>(a.location = r.destination)
}

fact noSharedTS{
    no disjoint u1,u2:User| some ts:TimeSlot|
        (ts in (u1.hasTimeSlot & u2.hasTimeSlot))
}

// each subroute belong to a route means this subroute is in its subroute
fact belongtoInverseSubroute{
    all r:Route, sub:SubRoute|
        (sub in r.subRoute => sub.belongTo = r)
    and
        (sub.belongTo = r => sub in r.subRoute)
}

//each subroute time sum up equals total time on the route
fact sumUpTimeOnTheRoad{
    all r:Route|
        (ts in (u1.hasTimeSlot & u2.hasTimeSlot))
}

// each subroute belong to a route means this subroute is in its subroute
fact belongtoInverseSubroute{
    all r:Route, sub:SubRoute|
        (sub in r.subRoute => sub.belongTo = r)
    and
        (sub.belongTo = r => sub in r.subRoute)
}

//each subroute time sum up equals total time on the route
fact sumUpTimeOnTheRoad{
    all r:Route|
     sum(r.subRoute.subTime)=r.totalTime
}

//each appointment should at least have one path
fact atLeastOnePath{
    all a1:RegisteredAppointment|#a1.route>0
}

//the route should now show the traffic method which is not in the user preference

fact fulfillTrafficPreference{
    all u:User,sub:SubRoute|
        ( sub in u.hasAppointment.route.subRoute) => (sub.trafficMethod in u.hasSettings)
}
```

6.3 Dynamic Models



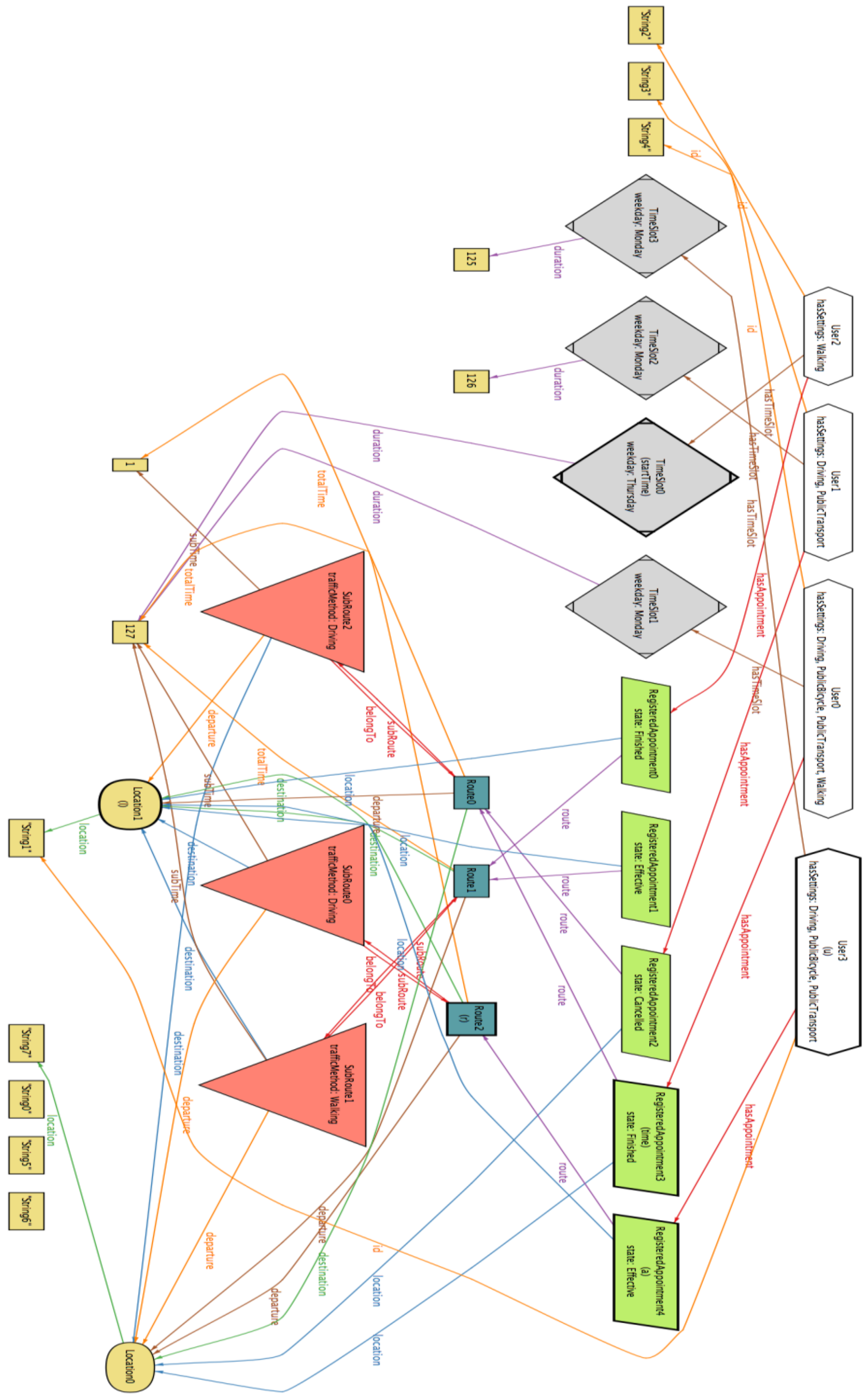
```
////////////PREDICATES////////////

pred ReachDestination[r:Route,l:Location]{
    r.destination = l
}


pred AddUserHasAppointment[u,u1:User, a:RegisteredAppointment]{
    u.hasAppointment=u1.hasAppointment+a
}

pred AddUserHasPreference[u,u1:User, tm:TrafficMethods]{
    u.hasSettings=u1.hasSettings+tm
}

pred AddUserHasTimeSlot[u,u1:User, ts:TimeSlot]{
    u.hasTimeSlot=u1.hasTimeSlot+ts
}
```



```
pred AddUserHasAppointment[u,u1:User, a:RegisteredAppointment]{
    u.hasAppointment=u1.hasAppointment+a
}

pred AddUserHasPreference[u,u1:User, tm:TrafficMethods]{
    u.hasSettings=u1.hasSettings+tm
}

pred AddUserHasTimeSlot[u,u1:User, ts:TimeSlot]{
    u.hasTimeSlot=u1.hasTimeSlot+ts
}
```

```
pred MakeAppointment[u:User,a:RegisteredAppointment, l:Location, r:Route,t :Time]{
    AddUserHasAppointment[u,u,a]
    a.location = l
    a.route = r
    a.time =t
    a.state = Effective
}

pred MakeTimeSlot[u:User,ts:TimeSlot, t1,t2:Time,dur:Int,d:Weekdays]{
    AddUserHasTimeSlot[u,u,ts]
    ts.startTime = t1
    ts.endTime = t2
    ts.duration = dur
    ts.weekday = d
}

pred CancelAppointment[a:RegisteredAppointment]{
    a.state = Cancelled
}

pred FinishAppointment[a:RegisteredAppointment, l:Location]{
    ReachDestination[a.route,l]
    a.state = Finished
}

pred show {
// #User=1
// #RegisteredAppointment=1
}

run MakeAppointment for 5 but 8 Int, exactly 8 String
run CancelAppointment  for 5 but 8 Int, exactly 5 String
run FinishAppointment  for 5 but 8 Int, exactly 5 String
run MakeTimeSlot  for 5 but 8 Int, exactly 5 String
run AddUserHasPreference for 5 but 8 int, exactly 5 String
```

6.4 Model Diagram



"String2"
"String3"
"String4"
TimeSlot3 weekday: Monday
TimeSlot2 weekday: Monday
TimeSlot0 (startTime) weekday: Thursday
TimeSlot1 weekday: Monday
125
126
1
127
User2 hasSettings: Walking
User1 hasSettings: Driving, PublicTransport
User0 hasSettings: Driving, PublicBicycle, PublicTransport, Walking
User3 (u) hasSettings: Driving, PublicBicycle, PublicTransport
SubRoute2 trafficMethod: Driving
SubRoute0 trafficMethod: Driving
SubRoute1 trafficMethod: Walking
Route0
Route1
Route2 (r)
RegisteredAppointment0 state: Finished
RegisteredAppointment1 state: Effective
RegisteredAppointment2 state: Cancelled
RegisteredAppointment3 (time) state: Finished
RegisteredAppointment4 (a) state: Effective
Location1 (l)
Location0
"String1"
"String7"
"String0"
"String5"
"String6"
id
duration
hasTimeSlot
hasAppointment
totalTime
subTime
departure
destination
location
belongTo
subRoute
route

7. Appendix

7.1 Used Tools

- **Microsoft Office 365 Word**
- **Alloy**
- **StarUML**
- **Microsoft Office 365 PowerPoint**

7.2 Hours of Works

7.2.1 CHANG LIN

Date	**Task**	**Hours**
8/10/2017	Project discussion	2
11/10/2017	Goals	2
15/10/2017	Product functions	2
17/10/2017	Dependencies	2
20/10/2017	Use case description and diagram	4
21/10/2017	Class diagram and sequence diagram	4
23/10/2017	RASD	1
28/10/2017	Final modification	2

7.2.2 MINGJU LI

Date	**Task**	**Hours**
8/10/2017	Project discussion	2
11/10/2017	Introduction	1
15/10/2017	Domain assumptions	3
16/10/2017	User interfaces	2
20/10/2017	Design constraints	1
21/10/2017	Scenarios	3
23/10/2017	ALLOY	5
28/10/2017	Final modification	2