Politecnico di Milano

AA 2017-2018

Computer Science and Engineering

Software Engineering 2 Project

TRAVLENDAR+

RASD

Requirement Analysis and Specification Document

Spoto Marco 829633

Table of Contents

Chapter 1

Introduction

1.1 Purpose

This is the Requirement Analysis and Specification Document (RASD). Through this document I want to describe Travlendar+, that is a calendar-based application that analyzes needs of the users in order to plan their activity, in terms of functional and non-functional requirements. I also want to show the constraints and the limits of the application. The document is created for all people who want to understand how the application is built and all the aspects about it.

- [G1] Allow an User to become registered after providing credential and accepting all terms.
- [G2] Allow an User to view the calendar.
- [G3] Allow an User to create/delate an appointment.
- [G4] Allow an User to change his credentials, to add a credit card (used for pay automatically) and to see all the information about an alert.
- [G5] Allow an User to choose any type of preferences.
- [G6] Offer the user an option to arrive on time to his/hers aoppointment.
- [G7] Allow a Security System to avoid any problem.
  - [7.1] Allow a Security System to give an alert if new appointment is impossible to plan.
  - [7.2] Allow a Security System for insured the secrecy of the user's data and plan.

1.2 Scope

Travlendar+ is a calendar-based application intended to program user's day. Obviously, the system should give the same functionalities of a normal calendar, like create/delate appointment.

Travlendar+ has the intent to take all user's plan and organize all kind of aspects for them, giving the better solution of journey from one appointment

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- *System Manager:* it is used to speck about the software implementation. It has the algorithm who pick up all the information from the external application and collect them to give a solution to the User.
- *Pay with a credit card:* you can pay with a credit card only if you add it in your credential. The payment management is managed by an external application. If the payment has something wrong the security system generate an alert; through this you can retry the payment or decide to pay autonomously. (N.B. if you add a credit card all payment that can be made through it are automatically done).
- *Asterisk:* means that it is not a mandatory information.
- *Application:* by this term I refer to the web application.
- *Preferences:* are all the alternative, about means of transport, among which the user can chose: car bike motorcycle, foot, taxi, uber, public means, plane.

1.3.2 Acronyms

- RASD: Requirement Analysis and Specification Document.
- API: application program interface

1.3.3 Abbreviations

- [Gn]: n-goal.
- [Dn]: n-functional requirement

1.4 Reference Documents

- The project description document: Mandatory Project Assignments (RASD and DD).pdf
- Integration tasting example document.pdf
- Some slides of the course.pdf
- http://alloy.mit.edu/alloy/tutorials/online/frame-FS-0.html.

1.5 Document Structure

This document is composed by five parts:

1. In the first part we have an introduction to the problem, listing all the identified goals and providing some base information in order to better understand the order sections of the document.
2. The second part consist of an overall description of the system in which boundaries are identified, and the actors involved in the system's usage are listed. The boundaries are listed giving all the necessary assumption.
3. The third part is composed by the specific requirements identified, both functional and non-functional. It is also composed by UMML diagrams that model the system in details.
4. In the fourth part we have a list of different scenarios. Each of them describes a particular situation with the system might to cope with.
5. 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.

Chapter 2

Overall description

2.1 Product perspective

The application will be created to improve planning, maximizing efficiency in time usage. This will be possible because the application is online so is continuously update and can estimate new solution, instantly, to avoid something unexpected.

2.2 Product functions

In this section we are going to define a specific list of requirements that the application uses to reach the goal. We report the number of the goal that we have already written in the section of the Scope and then the requirements it must have to done to work correctly.

-[G1] Allow an User to become registered after providing credential and accepting all terms.

1. People must be able to create a new profile and to insert all data all required to became a user.
2. User must be able to edit his/hers personal data.
3. User must be able to recover his/hers password in order to do not lose permanently the access to the application.

-[G2] Allow an User to view the calendar & [G3] Allow an User to create/delate an appointment.

1. User must be able to see all the appointment he/she has planned.
2. User must be able to add an appointment inserting Title, starting time and the duration of it.
3. User must be able to delate an appointment created before.

-[G4] Allow an User to change his credentials and to add a credit card used for pay automatically.

1. User must be able to change his/hers credentials.
2. User must be able to add a credit card for the automatic payment.

-[G5] Allow an User to choose any type of preferences.

1. User can choose which kind of transport he/she prefers to use.
2. User must be able to change his/hers preferences.

3. User must be able to see all alternative of travel.
4. User must be able to know the costs and the time of the alternatives.

-[G6] Allow a System Manager to do operations on the system for updating and maintenance.

1. System must be able to update continuously the information according weather, traffic, incidents, ….
2. System must be able to switch trip's solution according user request.
3. System must send a reminder 15 minutes before any appointment
4. System must be able to pay all kind of public transport trough an external Payment System (if the user has added a Credit card).
5. System must be sure you will never be late.

-[G7] Allow a Security System to avoid any problem.

1. Security system must be able to send an alert if there is an overlap of appointment or if it is impossible to arrive in time to one event.
2. Security system must guarantee privacy of any kind of data.

2.3 User characteristic

Travlendar+ can be used from all kind of people that want to implement their plan without any lose of time. It is created principally for business men that have to move from one place to another very frequently during the day.

2.3.1 Actors

- **User:** a person passed to a successful registration and how accept all terms and condition, able to use Travlendar+. He/she can login to the application and use all the application's functionalities.
- **System Manager:** a software able to collect all data and update the application giving all possible solution and selecting the best one.
- **Security System:** a software able to control the privacy of the user's data and the possibility of create new appoint without any overlaps or the impossibility to arrive on time.
- **Means of Transport:** ATM/OFO/MOBIKE/TRAIN/PLANE /CAR/MOTOCYCLE/TAXI transport used by the application to move users
- **Waze:** estimated time of trip.

- **Google Maps:** estimate time of rip.

2.4 Constraints

2.4.1Regularity policies

The system will have to ask for users' permission in order to connect mobile calendar to Travlendar+ and to use their position. The system will ask for users' payment informations and obviously, in addition to store them safely, will use them only for fees and rides payments.

2.4.2 Hardware limitations

- Mobile app
  - iOS, Android and Windows smartphone
  - 2G/ 3G/ 4G/ Wi-fi connection
  - GPS

You can connect yours mobile app with the web app.

2.5 Assumptions and dependencies

This part is written in order to better clarify some situation trough introduce the following assumptions.

2.5.1 Text assumptions

- People have to provide name, surname, address, telephone number, email, driving license* username and password to become a registered user. To complete the registration, they have also to confirm their registration.
- Password has to contain at least 8 characters and one of that must be a number in order to guarantee more security.
- In order to access in the application, the user has to insert telephone number or username and him/her password. He/she can also decide to keep the session active or save credentials.
- The amount of time and the price to move from one to another appointment can be valued from Waze, in case we use a car or go on foot or by bicycle, or by Google Maps, in case we use public transport.
- Appointment may be added and delated to the calendar.

- The user can choose which kind of transport he prefers that could be a combination of different.
- Security system allows an appointment if and only if it is possible for the user to arrive on time.
- If the user add a Credit card every possible payment are done with it.
- When weather is not good and you have already chosen an alternative like bicycle or motorcycle Security System provide an alert that suggest you to change your journey' alternative.

2.5.2 Domain assumptions

- [D1] Payment have to be verified by external service that will be in charge of all the payment process.
- [D2] If the payment, for some reason, cannot be completed the Security system creates an alarm that the user can delete in order to retry payment or decide to pay autonomously.
- [D3] The username must be unique.
- [D4] Users and any kind of sharing transport locations are retrieved by a GPS.
- [D5] Users fits all appointment in the correct timetables.
- [D6] When the system manager give the route time it means that at that moment and in that condition it is the journey time.
- [D7] It is impossible to do not have any kind of route alternative to arrive on time at the appointment.
- [D8] The weather forecast given from a external application are considered correct.
- [D9] The application is connected to car and bike sharing applications in order to use these.

Chapter 3

Specific requirements

3.1 External interface requirements

3.1.1 Software interfaces

- Google Maps and Waze APIs are needed to compute the estimated time it would take to get from one place to another.
- Car and bike sharing services APIs are necessary to estimate all alternative routes.
- Weather services APIs is necessary to considerate weather conditions.

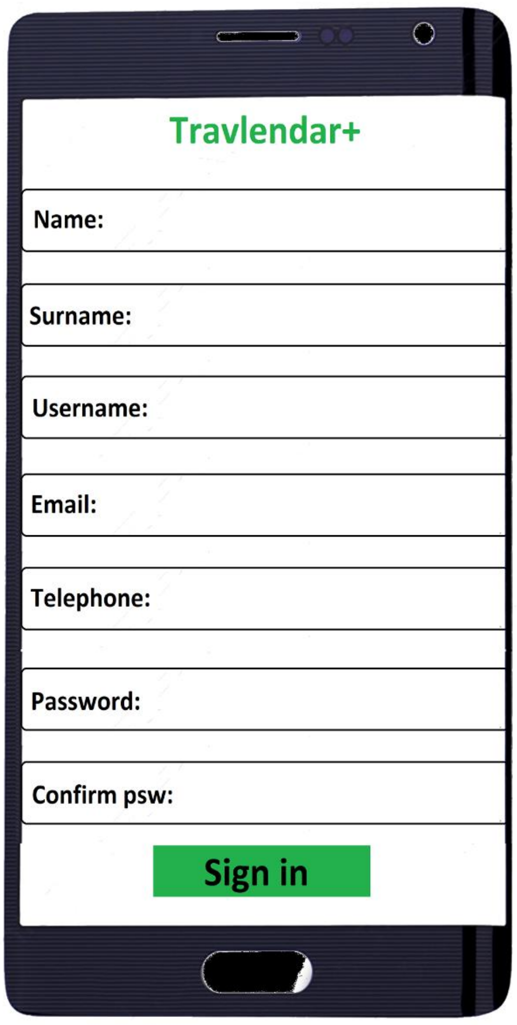
3.1.1 User interfaces

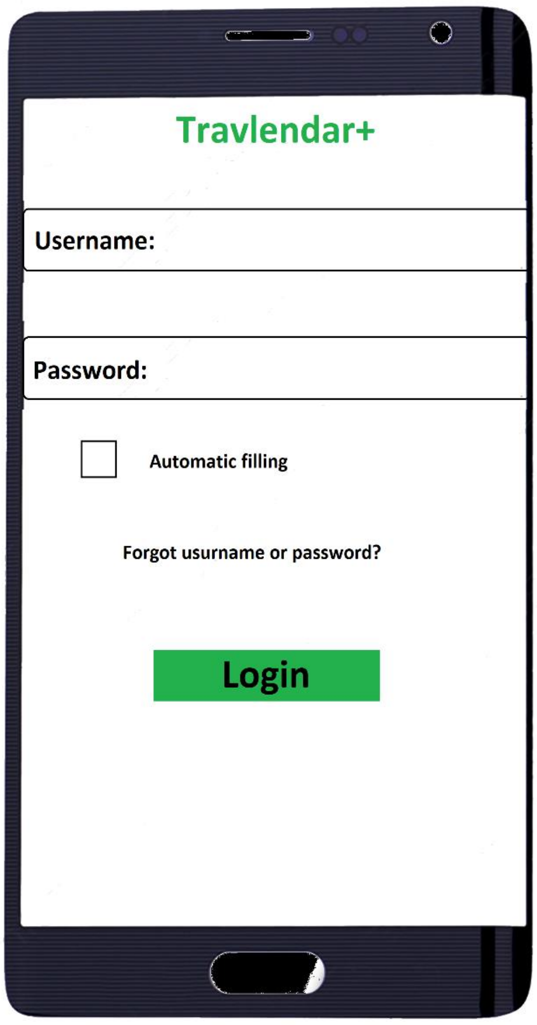
This application take cue from lot of other one in order to simplify to the User the use of this app. He/she can learn in very few passages how to use this app.

Here below the basic idea what the mobile app will look like.

- ✓ Sign in

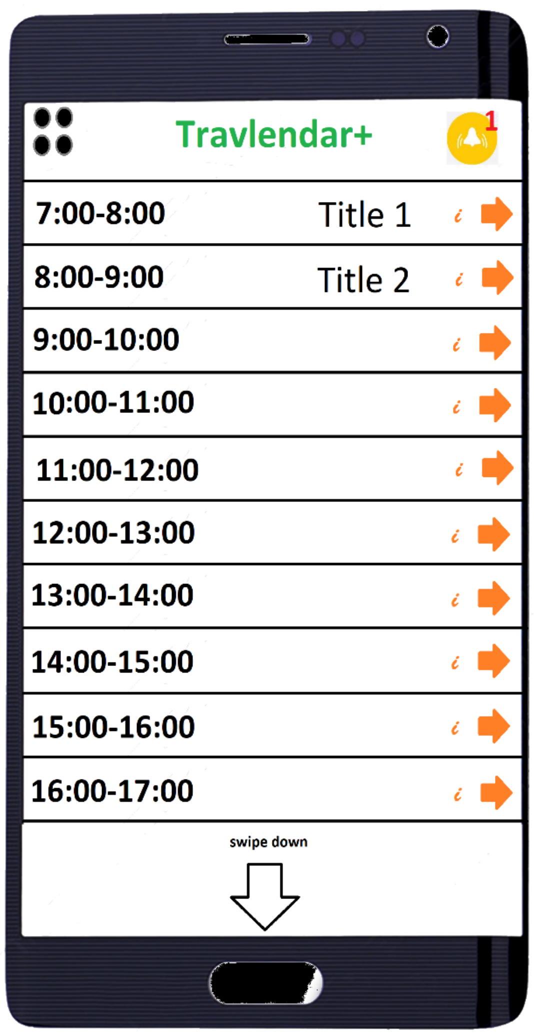
Login





✓ General view of the day

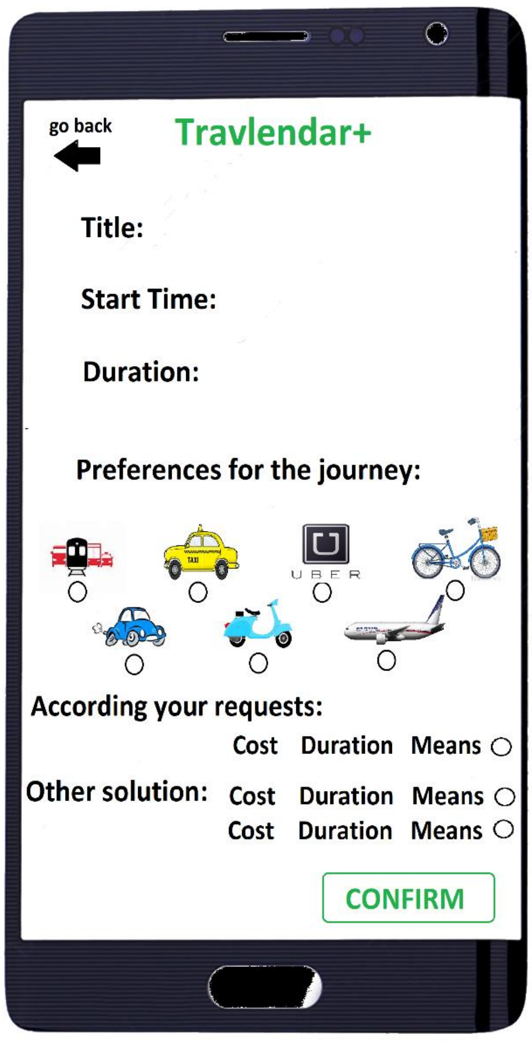
By clicking "*i*" in an free timeslot



Travlendar+ 1

7:00-8:00	Title 1	*i*	➡
8:00-9:00	Title 2	*i*	➡
9:00-10:00		*i*	➡
10:00-11:00		*i*	➡
11:00-12:00		*i*	➡
12:00-13:00		*i*	➡
13:00-14:00		*i*	➡
14:00-15:00		*i*	➡
15:00-16:00		*i*	➡
16:00-17:00		*i*	➡

swipe down



go back

Travlendar+

Title:

Start Time:

Duration:

Preferences for the journey:

○ ○ ○ ○ ○ ○ ○

According your requests:
Cost Duration Means ○

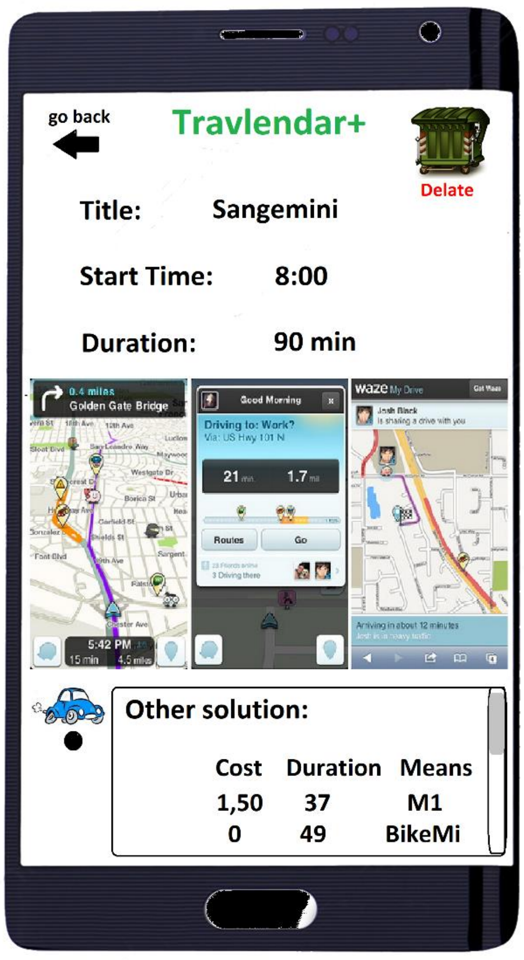
Other solution: Cost Duration Means ○
Cost Duration Means ○

CONFIRM

✓ By clicking on the bell:

- If there are some notifications, indicated by the number at the top right, it open a curtain with the specification of the alarm.

- ✓ By clicking “i” in a busy timeslot



Travlendar+

Title: Sangemini

Start Time: 8:00

Duration: 90 min

Other solution:

Cost	Duration	Means
1,50	37	M1
0	49	BikeMi

- ✓ By clicking the 4 points on the top left:
  - it open a curtain in which you can modify all the information about you or change password.
  - Here you have also the possibility to add your credit card to give the possibility at the app to pay autonomously (if you add the CreditCard you authorized automatically the app to do all possible payment through an external system. If something go wrong an alert marks the problem).

3.2 Functional Requirements

3.2.1Use case descriptions

3.2.1.1 Visitor registration

ACTORS	Visitor
GOALS	[G1]
INPUT CONDITION	There are no entry conditions
EVENTS FLOW	a. The visitor open the application downloaded from the app store. b. The visitor fills all the mandatory files. c. The visitor clicks on the "Sign in" button. d. The system send an email to the visitor to confer the registration. e. The visitor confirm the sign in by clicking on the link in the email. f. The system saves the data.
OUTOUT CONDITIONS	The visitor successfully ends the registration process and become a new user. From now he/she can log in the application providing his/hers credentials and start using travlendar+.
EXCEPTIONS	a. The visitor is already a user. b. The user chooses a username that is already token. c. The user chooses a password that do not respect the requirements. d. The user inserts an not valid email. All the exception are handled notifying the issue to the visitor and squaring in red pans in which there is some of this problem.

3.2.1.2 Visitor login

ACTORS	User
GOALS	[G1]
INPUT CONDITION	The User has already done the registration
EVENTS FLOW	a. The user inserts his/hers credentials into the "username" and "password" fields if the automatic fill is not selected. b. The user show to the user the general view of the day .
OUTOUT CONDITIONS	The user can see all day appointment.
EXCEPTIONS	a. The user inserts a not valid username. b. The user inserts a not valid password. All the exception are handled notifying the issue to the visitor and squaring in red pans in which there is some of this problem.

3.2.1.3 User view and modify calendar

ACTORS	User
GOALS	[G2] &[G3]
INPUT CONDITION	The user has to be already logged into the system.
EVENTS FLOW	a. The user looks at all already existing appointment's time and title. b. The user click "*i*" or to create a new event if the time slot is empty; or to delate an event; or to get some information about that event (selected preferences, duration, costs, alternative of journey). c. The user can change his/hers preferences.
OUTOUT CONDITIONS	The user has successfully look at the calendar adding or delating his/hers appointment, modify his/hers credentials.
EXCEPTIONS	a. The user create an event that is unattainable in the time required. This exception is handled redirecting the User to an alert in which will be explained the issue occurred.

3.2.1.3.1 User creates new event

ACTORS	User
GOALS	[G3]
INPUT CONDITION	The user has to be already logged and click "*i*" in a slot of time without any event.
EVENTS FLOW	a. The user inserts a title at the event b. The user adds a start time c. The user adds the duration d. The user selects the preferences e. The user chooses trough the alternative
OUTOUT CONDITIONS	The user has successfully created an event
EXCEPTIONS	a. The user overlap two events b. The user take an appointment in a slot of time that he/she cannot reach in time This exception is handled redirecting the User to an alert in which will be explained the issue occurred.

3.2.1.3.2 User delate an event

ACTORS	User
GOALS	[G3]
INPUT CONDITION	The user has to be already logged and click "*i*" in a slot of time with an event.
EVENTS FLOW	a. The user clicks in the creel on the top right.
OUTOUT CONDITIONS	The user has successfully delated an event
EXCEPTIONS	No exceptions

3.2.1.3.3 User change preferences for the journey's event

ACTORS	User
GOALS	[G3]
INPUT CONDITION	The user has to be already logged and click "*i*" in a slot of time with an event.
EVENTS FLOW	a. The user clicks on a different kind of preferences. b. The user chooses a different alternative for his/hers journey.
OUTOUT CONDITIONS	The user has successfully changed his/hers journey, knowing all about the new cost time and means.
EXCEPTIONS	No exceptions

3.2.1.4 User view and modify his credentials

ACTORS	User
GOALS	[G4]
INPUT CONDITION	The user has to be already logged and clicked on the four points in the top left.
EVENTS FLOW	a. The user clicks on the credential he/she wants to change. b. The user change one or more filed
OUTOUT CONDITIONS	The user has successfully changed his/hers credentials
EXCEPTIONS	a. The user inserts a username or an email already token. The exception is handled notifying the issue to the visitor and squaring in red pans in which there is this problem.

3.2.1.5 User add a credit card

ACTORS	User
GOALS	[G4]
INPUT CONDITION	The user has to be already logged and clicked on the four points in the top left.
EVENTS FLOW	a. The user clicks on add a credit card. b. The user fills the fields for add the credit card and accept terms and conditions.
OUTOUT CONDITIONS	The user has successfully added his/hers credit card.
EXCEPTIONS	a. The user inserts a not valid data. The exception is handled notifying the issue to the visitor and squaring in red pans in which there is this problem.

3.2.1.6 User view all the alert

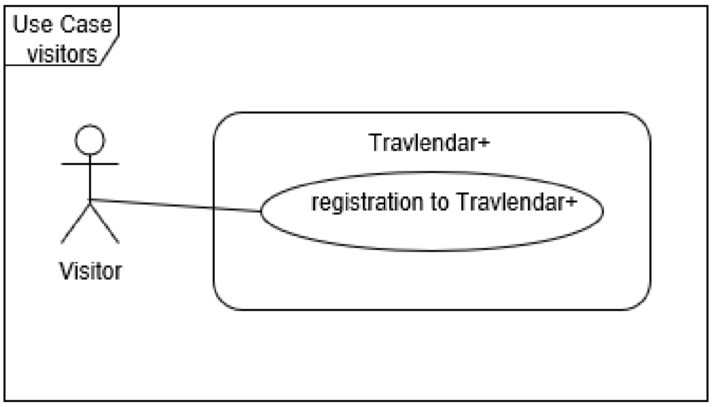
ACTORS	User
GOALS	[G4]
INPUT CONDITION	The user has to be already logged and clicked on the bell in the top right.
EVENTS FLOW	a. The user clicks on the title alarm of which he/she wonts more information.
OUTOUT CONDITIONS	The user know why there is an alert
EXCEPTIONS	No exception.

3.2.1.7 User choose preferences and alternatives

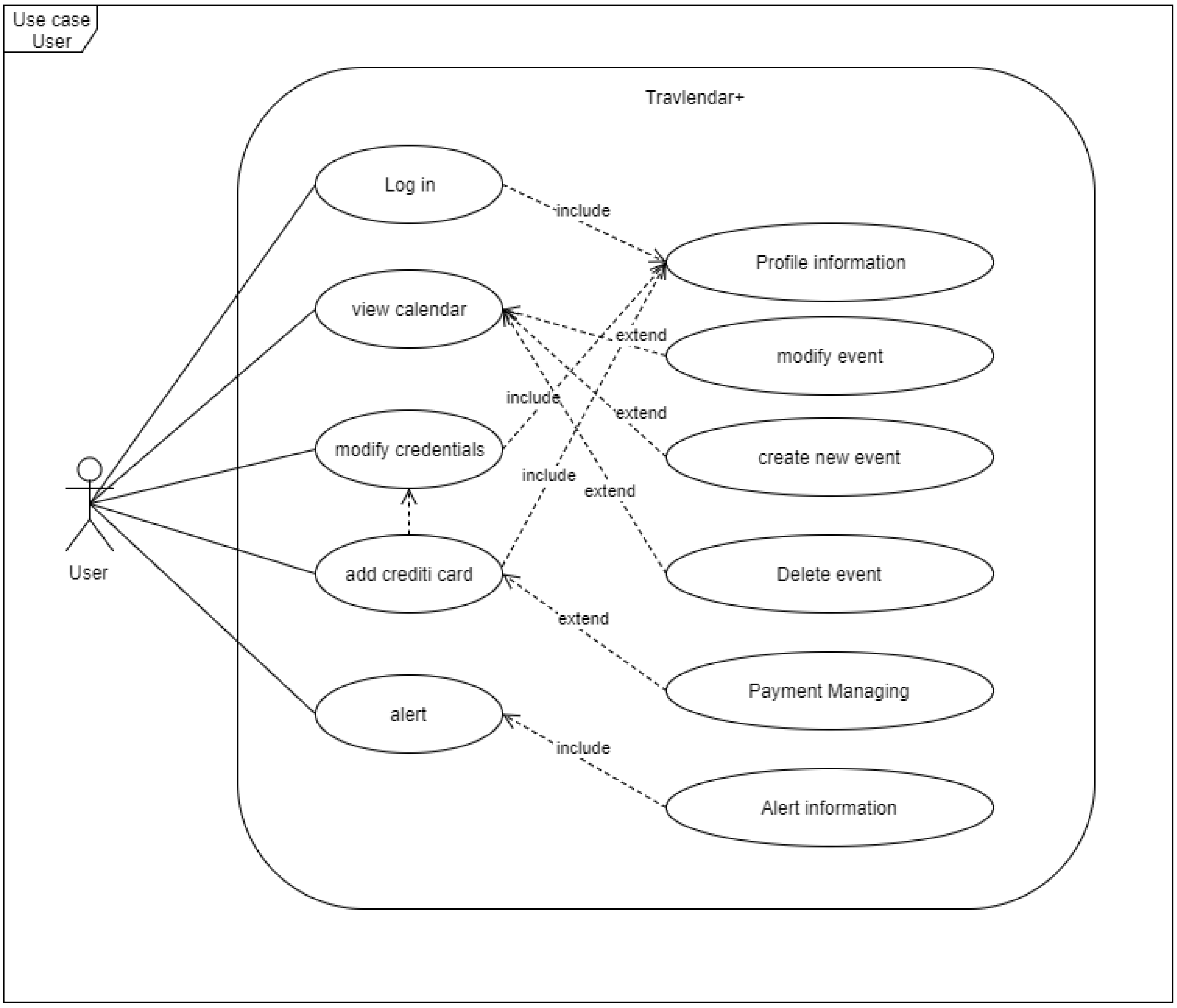
ACTORS	User
GOALS	[G5]
INPUT CONDITION	The user has to be already logged and clicked on the “i” of an event.
EVENTS FLOW	a. The user chooses the preferences that he/she wants. b. The user chooses the alternatives he/she wants.
OUTOUT CONDITIONS	The user has chosen the preferences and than the alternatives he/she prefers.
EXCEPTIONS	No exception.

3.2.2 Use case diagram

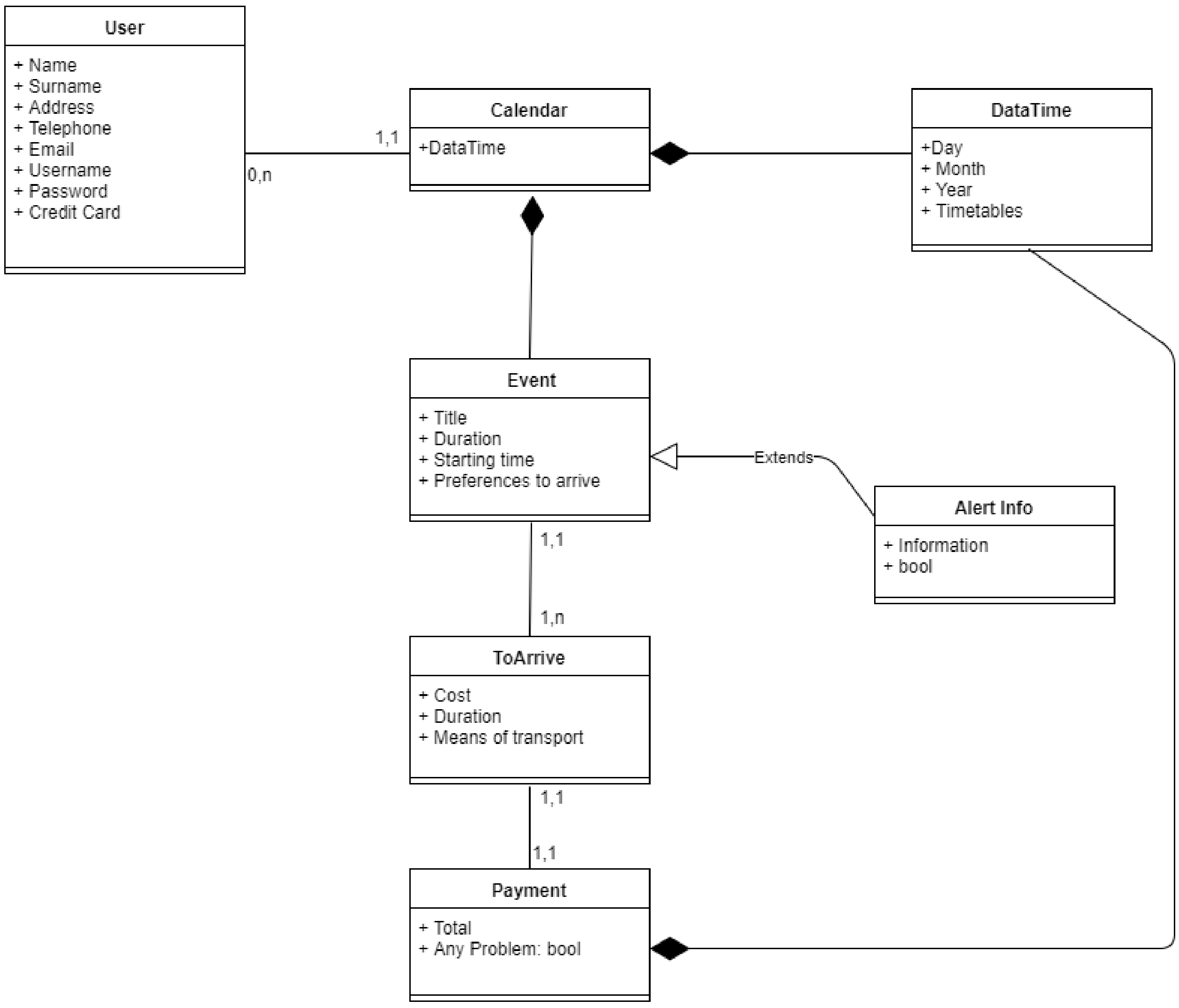
3.2.2.1



3.2.2.2

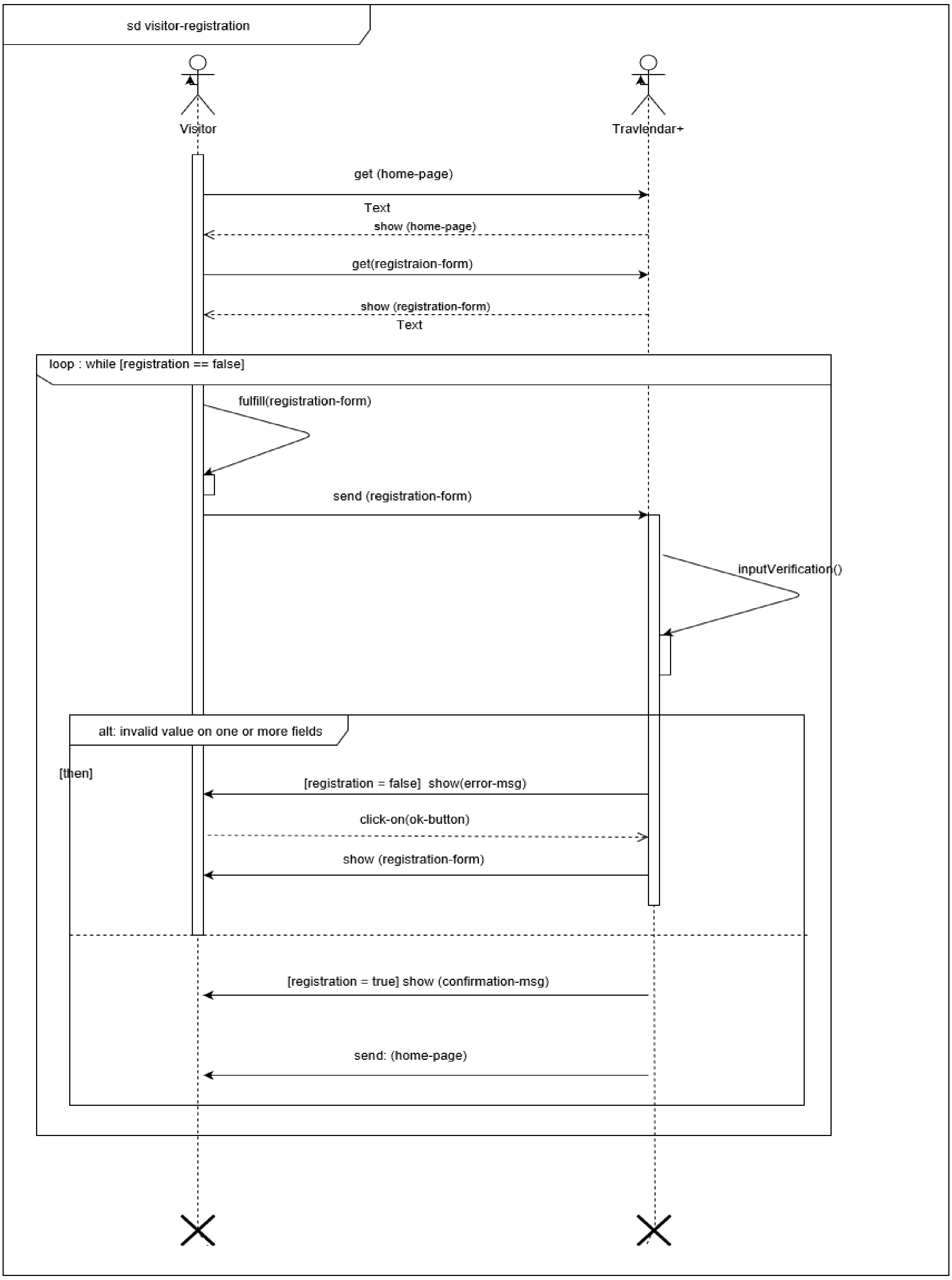

Use case User
Travlendar+
Log in
view calendar
modify credentials
add crediti card
alert
User
Profile information
modify event
create new event
Delete event
Payment Managing
Alert information
include
extend
include
extend
include
extend
extend
include

3.2.3 Class diagram

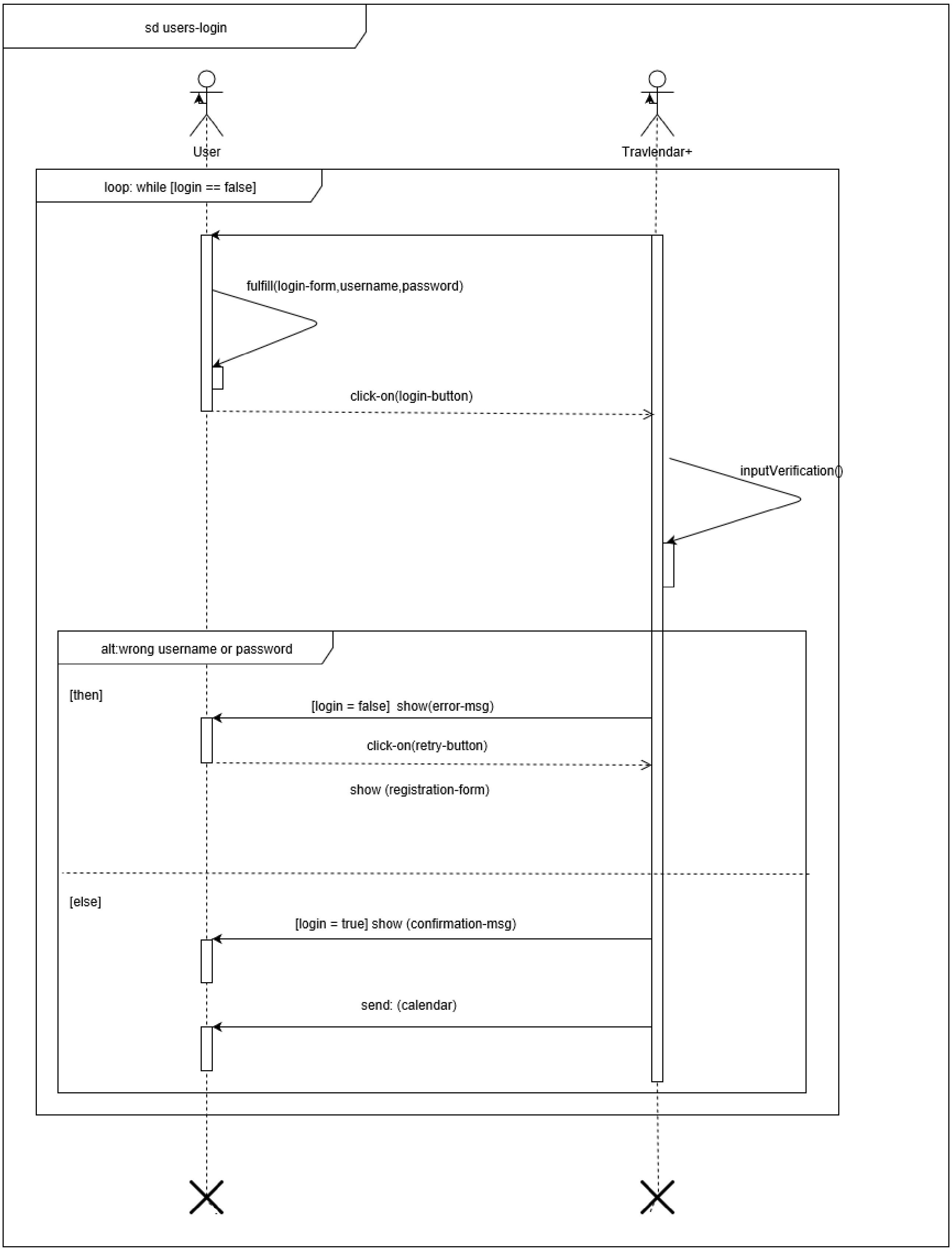


3.2.4 Sequence diagram

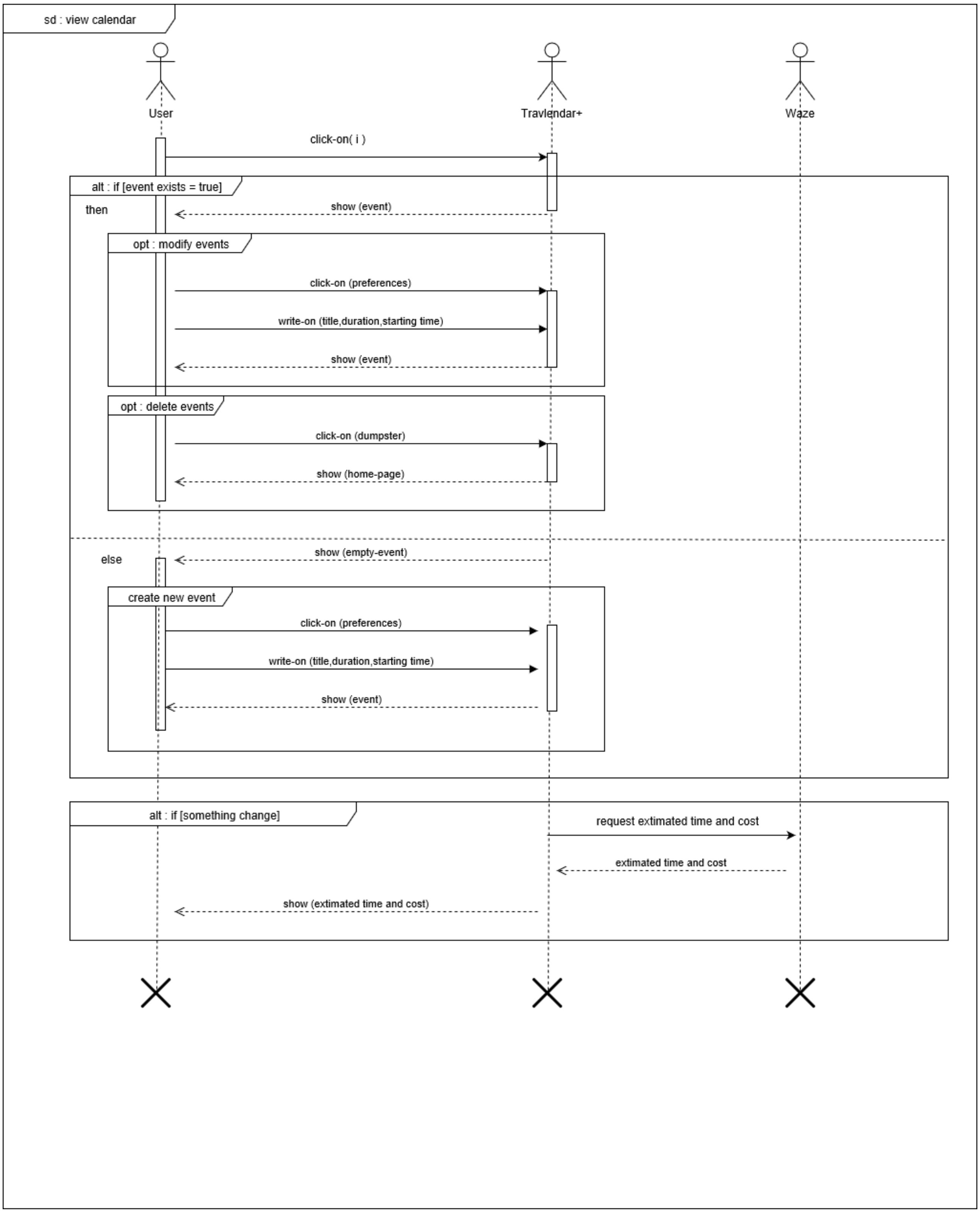
3.2.4.1



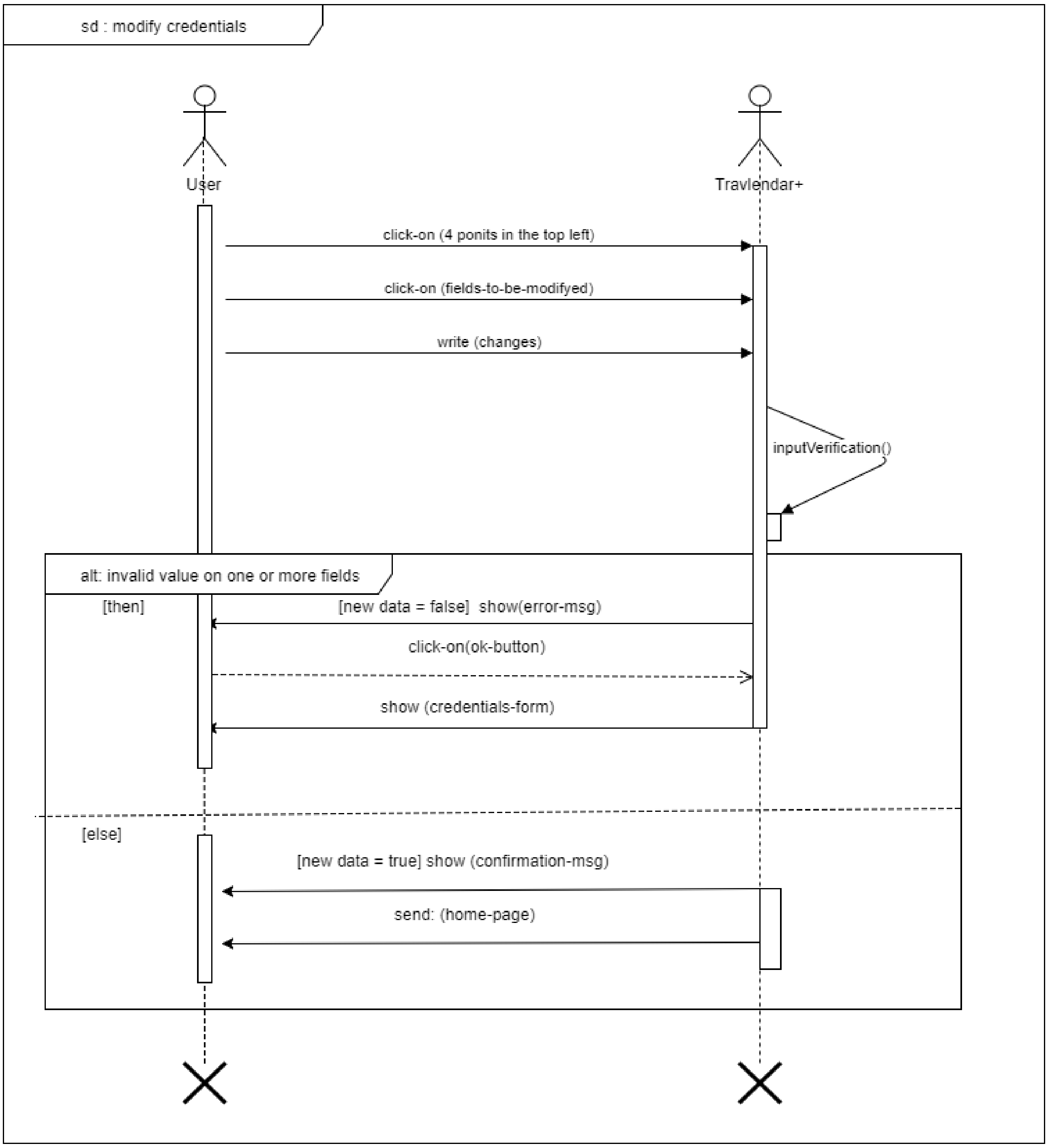
3.2.4.2


sd users-login
User
Travlendar+
loop: while [login == false]
fulfill(login-form,username,password)
click-on(login-button)
inputVerification()
alt:wrong username or password
[then]
[login = false] show(error-msg)
click-on(retry-button)
show (registration-form)
[else]
[login = true] show (confirmation-msg)
send: (calendar)

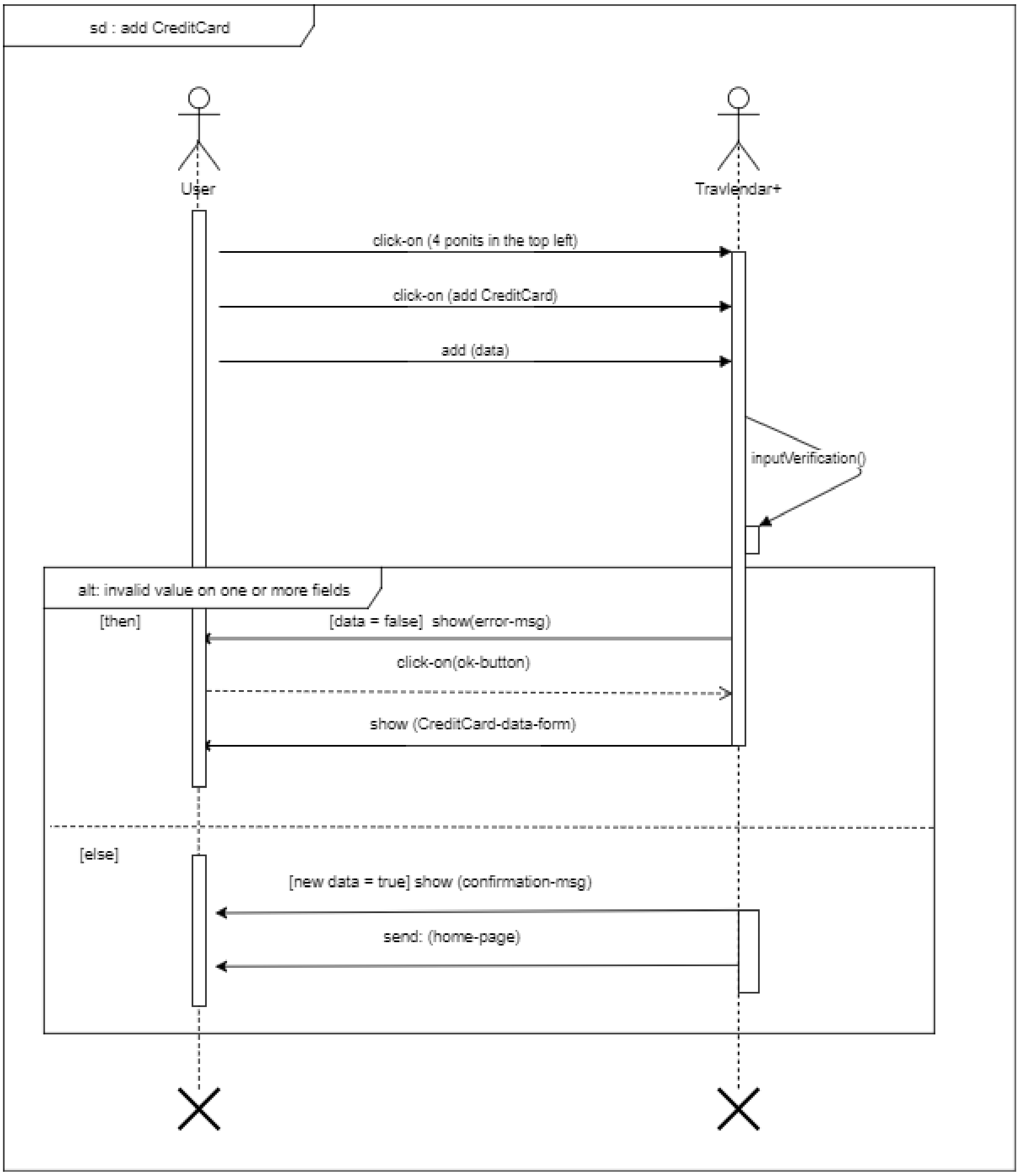
3.2.4.3


sd : view calendar
User
Travlendar+
Waze
click-on(i)
alt : if [event exists = true]
then
show (event)
opt : modify events
click-on (preferences)
write-on (title,duration,starting time)
show (event)
opt : delete events
click-on (dumpster)
show (home-page)
else
show (empty-event)
create new event
click-on (preferences)
write-on (title,duration,starting time)
show (event)
alt : if [something change]
request extimated time and cost
extimated time and cost
show (extimated time and cost)

3.2.4.4


sd : modify credentials
User
Travlendar+
click-on (4 ponits in the top left)
click-on (fields-to-be-modifyed)
write (changes)
inputVerification()
alt: invalid value on one or more fields
[then]
[new data = false] show(error-msg)
click-on(ok-button)
show (credentials-form)
[else]
[new data = true] show (confirmation-msg)
send: (home-page)

3.2.4.5


sd : add CreditCard
User
Travlendar+
click-on (4 ponits in the top left)
click-on (add CreditCard)
add (data)
inputVerification()
alt: invalid value on one or more fields
[then]
[data = false] show(error-msg)
click-on(ok-button)
show (CreditCard-data-form)
[else]
[new data = true] show (confirmation-msg)
send: (home-page)

3.3 Non-Functional Requirements

3.3.1 Performance

The system has to be able to have simultaneous calendars for each user. It must be able to compare all possible alternative and to choose the one that approaches more to the user preferences. It must be able to support a large number of users at the same time without stall. It must guarantee a very precise estimate of costs and duration of the journey.

3.3.2 Reliability

The system must guarantee a 24/7 service. Obviously we can accept very small deviations from this requirement for something like updates.

3.3.3 Security

Users credentials (and CreditCard data) will be stored. Security of the data and of all kind of appointment is a primary concern.

3.3.4 Scalability

Plan our day is very difficult mostly if we must move from one place to another lot of time during a day. This application is created principally to help this people so the system has to guarantee an high level of scalability both on cross maps information to have a more accurate estimate.

3.3.5 Accuracy

Accuracy of the information provided (position of available sharing, time of journey,.....) has to be the best possible. All the information must to provide a information with an error loss or equal to 5%.

Chapter 4
Scenarios

4.1 Scenario 1

Jason is an important business man who lives in Milan, he has a private studio but often must go out for work in Rome. For this reason he become an Travlendar+ user, in this way he can organize his trip according costs a time. The are two different type of travel to reach Rome from Milan; the first one is by plane an the second one is by train, obviously they are different; he prefer travel by plane because it is faster so he selects plane as preferences and the application find for him the plane to arrive in time ate the appointment. Jason has also added his credit card so when he chooses the alternative he wonts the application by his ticket and sends him an email with it and the only action that Jason must do is to do check-in.

4.2 Scenario 2

Mary is a University student. She wants to plan hers day according lessons, sport and study. She create all the events of the day and select hers preferences (obviously if she selects a plane like means of transport the application do not considers it). The system must guarantee no kind of overlap and must alert her when some event is unreachable. She select foot and bike-shearing as preferences so the application provide better solution according this preferences but if weather is not good the app create an alert to suggest her to choses another alternative of transport.

4.3 Scenario 3

Sally need to go to pick up some food at the supermarket for dinner and it is very late, she inserts the event in the calendar and does not select any preferences in this way the application give the faster solution to reach the supermarket without considering any constraints.

Chapter 5

Alloy

5.1 Signatures

```
open util/boolean


sig Stringa{}

sig Set{}

one sig User{
    calendar: some Calendar,
     username: one Stringa,
    usPosition : Position,
    preferences : one Stringa
}

sig Calendar{
     dataTime: DataTime,
    setOfEvents : set Event
}

sig DataTime{
    Data,Mounth: one Stringa,
    timeTables : TimeTables
}

some sig Event {
     busy: one Bool,
     dataTime:DataTime,
    Title: one Stringa,
    timeTables : TimeTables
}

sig TimeTables{
    startTime : one Time,
    endTime : one Time
}{
    startTime != endTime
}

sig Time{
    value: Int
}{
    value>0
}

sig AlertInfo extends Event{
    Information: one Stringa,
    Exist: one Int
}{
 all alert: Exist | alert >= 0 and alert <=10
}
```

```
sig ToArrive{
    startTime: one Time,
    endTime: one Time,
    startPos: Int,
    finalPos: Int,
    Cost: one  Int,
    TravelTime: one Time,
    MeansOfTransport: one Stringa
}{
    all c : Cost | c>0
     startPos != finalPos
}

sig Payment{
    relatedEvent: one Event,
    paymentIn : one DataTime,
    amount: one Int,
    completed: one Bool
}{
    all a : amount | a < 0
}

sig Position{
    latitude: Int,
    longitude: Int
}

abstract sig StopPosition extends Position { }
    sig StartPosition extends StopPosition { }
    sig FinalPosition extends StopPosition { }

sig Path {
  positions: seq StopPosition,
}
```

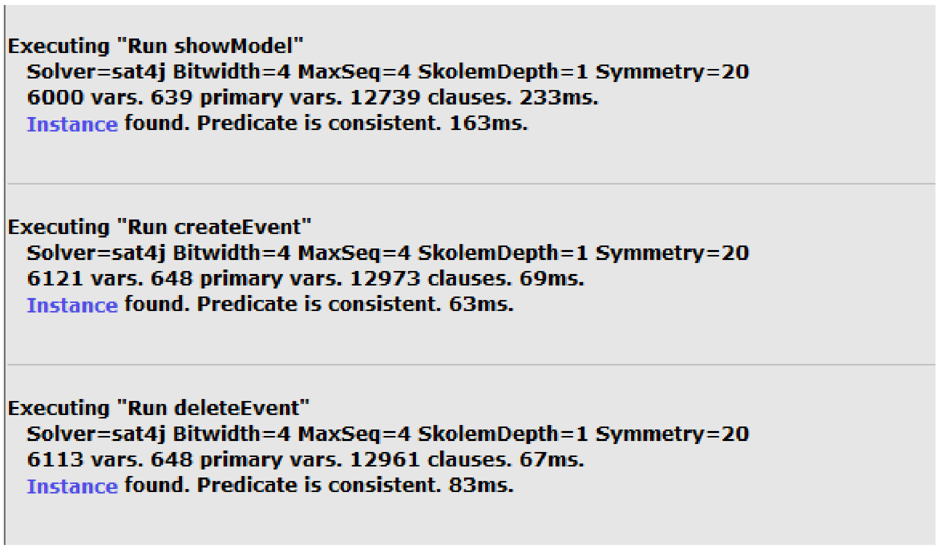
5.2 Facts

```
//FACTS

fact usernameIsUnique{
    no disjoint u1,u2: User| u1.username =u2.username
}
fact eventIsUnique{
    no disjoint e1,e2: Event | e1.dataTime = e2.dataTime
}
fact calendarDependFromUser{
    all c: Calendar, u :User | c in u.calendar
}

fact pathStartWithLoad {
    all p: Path | p.positions.first in StartPosition
}
fact pathEndWithDrop {
    all p: Path | p.positions.last in FinalPosition
}
fact noSameValuOfTimeslot{
    no disjoint t1,t2 : Time | t1.value = t2.value
}

fact noOverlaps{
    all disjoint e1,e2: Event | Booked [e1.timeTables.endTime, e2.timeTables.startTime] or Booked [e2.endTime, e1.startTime]
}
```

5.3 Predicates

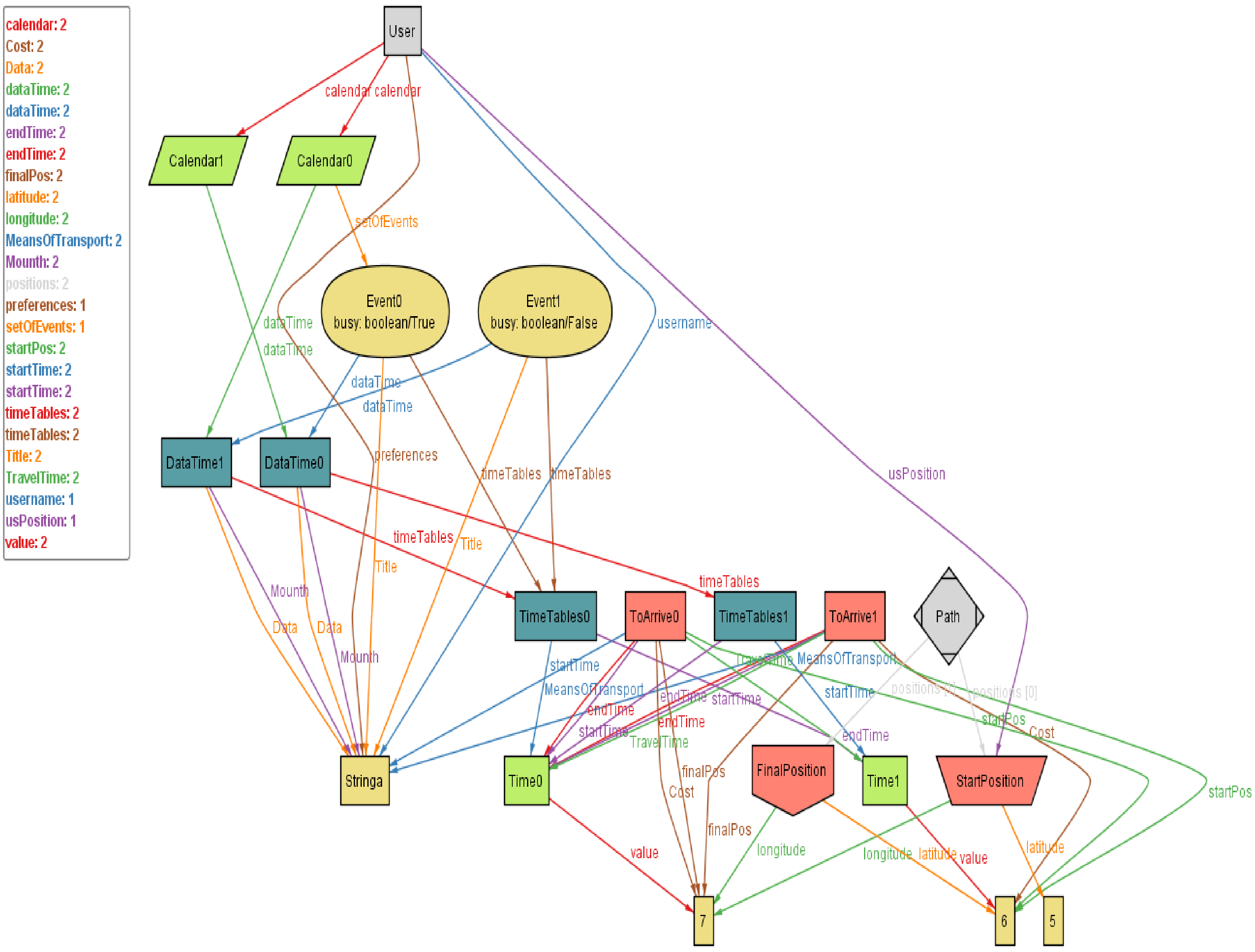
```
// DYNAMIC ANALYSIS
pred noOverlap{
    all disjoint e1,e2 : Event | Booked [e1.timeTables.endTime, e2.timeTables.startTime] or Booked [e2.timeTables.endTime, e1.timeTables.startTime]
}
pred Booked [t1,t2 : Time]{
    t1.value<t2.value
}
pred createEvent [e : Event, c,c' : Calendar]{
     e not in c.setOfEvents
    c != c'
    c'.setOfEvents =  (c.setOfEvents + e)
    noOverlap
}
pred deleteEvent [e : Event, c,c' : Calendar]{
     c != c'
    e in c.setOfEvents
    c'.setOfEvents = (c.setOfEvents - e )
}
pred showModel{}


//RESULTS

run showModel

run createEvent

run deleteEvent
```

5.4.1 Proof of consistency

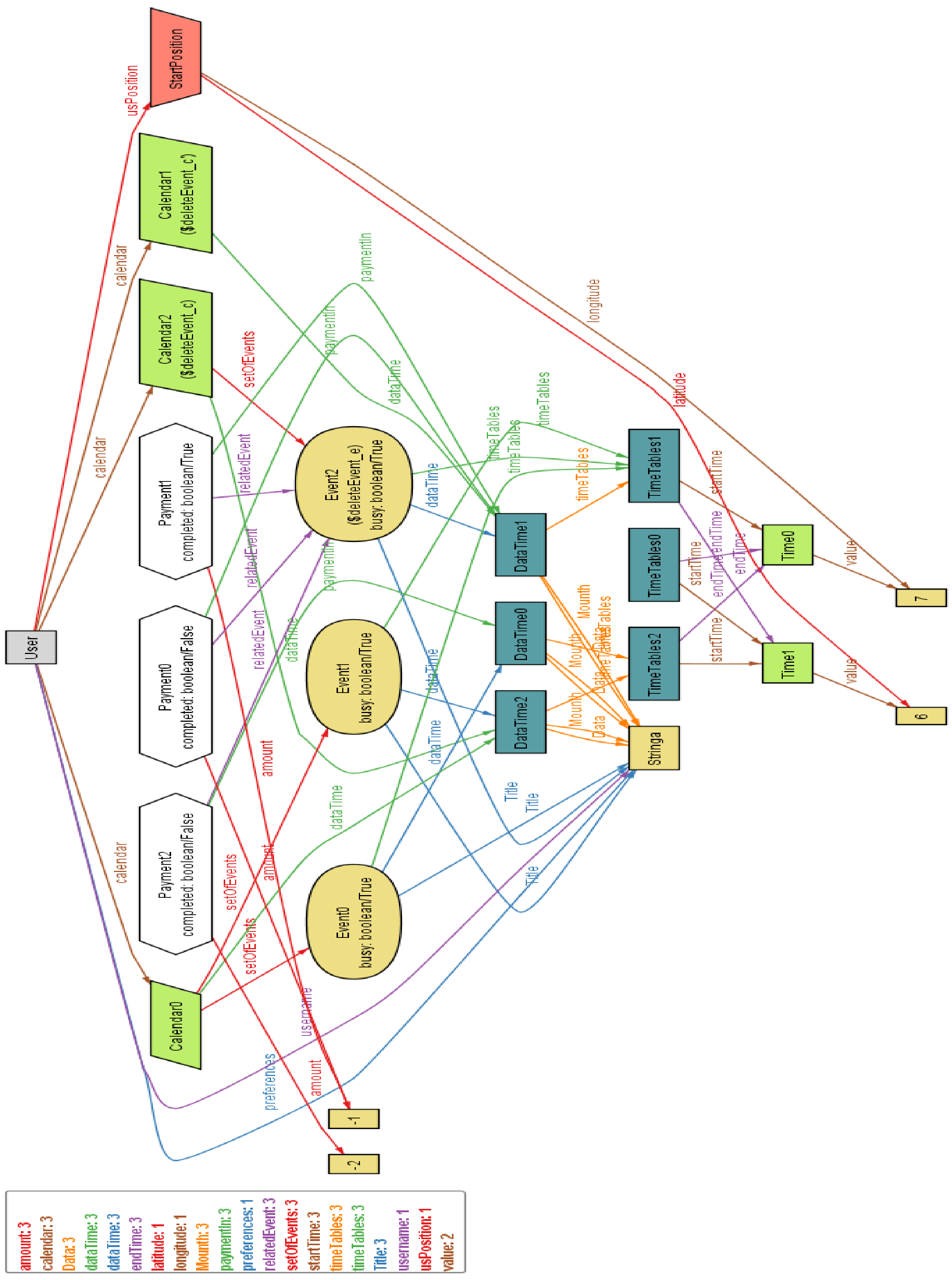


```
Executing "Run showModel"
   Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
   6000 vars. 639 primary vars. 12739 clauses. 233ms.
   Instance found. Predicate is consistent. 163ms.

Executing "Run createEvent"
   Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
   6121 vars. 648 primary vars. 12973 clauses. 69ms.
   Instance found. Predicate is consistent. 63ms.

Executing "Run deleteEvent"
   Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
   6113 vars. 648 primary vars. 12961 clauses. 67ms.
   Instance found. Predicate is consistent. 83ms.
```

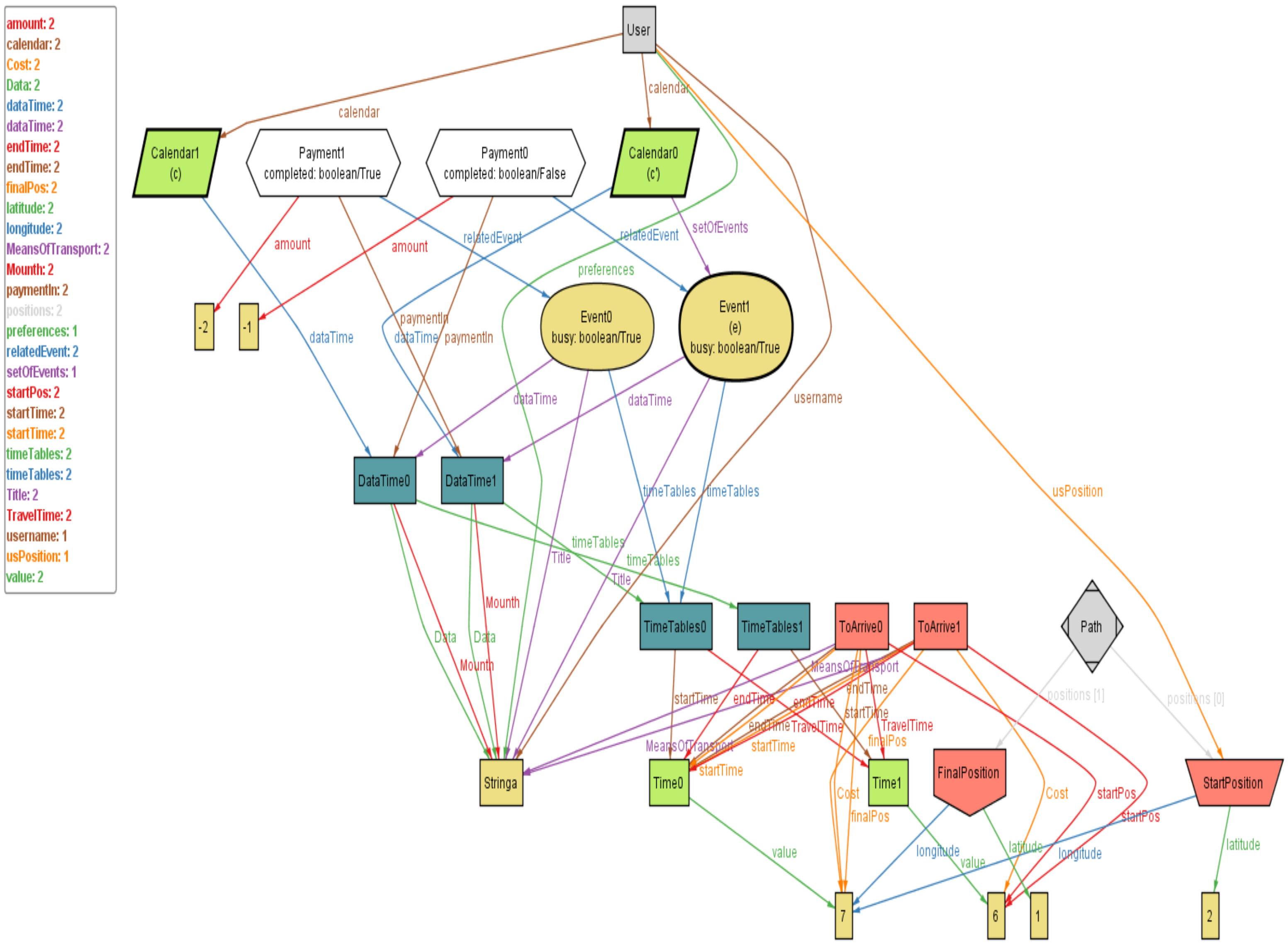
5.4.2 Generated world (showModel)



5.4.3 Generated world (deleteEvent)



5.4.3 Generated world (createEvent)



Chapter 6
Appendix

6.1 Revision history

DATE	TASK	HOURS
5/10/2017	Specifications and goals	3
7/10/2017	Goals	2
8/10/2017	Goals and assumptions	3
10/10/2017	Assumptions and Scenarios	3
12/10/2017	Requirements and Use cases diagrams	5
14/10/2017	Use case and Class diagrams	6
15/10/2017	Sequence diagrams	4
17/10/2017	Sequence diagrams	5
19/10/2017	Alloy	4,5
21/10/2017	Alloy	3,5
24/10/2017	Various	4
27/10/2017	RASD revision	6
28/10/2017	RASD and Alloy revision	7
	TOTAL	**56**

6.2 Used tools

- Drow io
- Alloy Analyzer 4.2
- Word