



Politecnico di Milano

Software engineering 2

myTaxiService

Requirements Analysis and Specifications Document (RASD)

790021 Antenucci Sebastiano
852461 Buonagurio Ilaria

Prof.ssa Di Nitto Elisabetta

Table of contents

<i>1 Introduction</i>	<i>4</i>
<i>1.1 Purpose</i>	<i>4</i>
<i>1.2 Actual system</i>	<i>4</i>
<i>1.3 Scope</i>	<i>4</i>
<i>1.4 Actors</i>	<i>5</i>
<i>1.5 Goals</i>	<i>5</i>
<i>1.6 Definitions, Acronyms, Abbreviations</i>	<i>6</i>
<i>1.6.1 Definitions</i>	<i>6</i>
<i>1.6.2 Acronyms</i>	<i>7</i>
<i>1.7 Reference documents</i>	<i>7</i>
<i>1.8 Document overview</i>	<i>8</i>
<i>2 General Description</i>	<i>10</i>
<i>2.1 Product context</i>	<i>10</i>
<i>2.1.1 System interface</i>	<i>10</i>
<i>2.1.2 User interface</i>	<i>10</i>
<i>2.1.3 Software interface</i>	<i>11</i>

2.1.4 Operations	12
2.2 Product functions	12
2.3 User characteristics	12
2.4 Constraints and assumptions	13
2.4.1 Hardware limitations	13
2.4.2 Interface to other applications	13
2.4.3 Requirement of high-level language	13
2.4.4 Reliability requirements	13
2.4.5 Security constraints	13
2.4.6 Other considerations	14
2.4.7 Assumptions	14
2.4.8 Future possible implementation	15
3 Specific requirements	17
3.1 External interface requirements	17
3.1.1 User interface	17
3.1.2 Taxi driver interface	30
3.2 Functional requirements	34
3.2.1 Unauthenticated user or visitor	34
3.2.2 Authenticated user	34
3.2.3 Unauthenticated taxi driver	35

3.2.4 <i>Authenticated taxi driver</i>	35
3.2.5 <i>Scenarios</i>	35
3.2.6 <i>Analysis model</i>	41
3.2.7 <i>Use case model</i>	42
3.2.8 <i>Use case description</i>	43
3.2.9 <i>Activity diagram</i>	51
3.3 <i>Non functional requirements</i>	53
3.3.1 <i>Reliability</i>	53
3.3.2 <i>Availability</i>	53
3.3.3 <i>Security</i>	53
3.3.4 <i>Maintainability</i>	54
3.3.5 <i>Portability</i>	54
3.4 <i>Design constraints</i>	54
4 <i>Appendix</i>	55
4.1 <i>Alloy</i>	55
4.1.1 <i>Diagrams</i>	56
4.1.2 <i>Code</i>	62
4.2 <i>Tools</i>	66
4.3 <i>Working time</i>	67

1 Introduction

1.1 Purpose

This document represents the Requirement Analysis and Specification Document (RASD), it helps the developers during the implementation phase, providing them the requirements specification that should be respected.

Its aim is to accurately and completely describe all the software's functional and non functional requirements and analyse the real need of the customer . In order to do so it contains an overall description and specification of the software, to define and model the project given from the stakeholder, it also includes a set of use case to describe the interactions the users will have with the software and simulate what will occur after the developement, in terms of what the customers, and so the software, is expected or not to do.

1.2 Actual System

At the actual state of thing there is a telephone number that can be called in order to reserve a taxi. When the user calls he receives information about the reservation number, waiting time and the taxi identification code.

At the same time the telephone operator notifies the driver about the destination, the number of passengers and the reservation number.

1.3 Scope

Foremost the aim of the project is to optimize the taxi service of a large city and in particular it is to simplify passengers' access to the service and to guarantee a fair management of the queues by creating a new software called myTaxiService.

Users can request a reservation through a web or mobile application and they are notified by the system with the code of the incoming taxi and the waiting time.

There are a two kind of reservation: the reservation, made at least two hours in advance of the meeting time by specifying the departure and arrival of the ride; or the “quick” one, by specifying only the origin of the ride. In the second case is supposed that the taxi is requested immediately.

In support to taxi drivers there is a mobile application that allows them to inform the system about their availability and they can also confirm or dismiss a certain call.

The system also simplifies queues management and taxi distribution on the territory, in order to make the service more efficient and to serve a larger number of passengers.

1.4 Actors

- Visitor: all visitor users can only see the login page and complete the registration form to be able to access to all the functionality of the application as registered user.
- Registered user: after a successful login on the system, this is the only kind of user that can reserve a taxi either in the quick way or in the programmed one. They will be advised about their reservation by the system.
- Taxi driver: after a successful login on a drivers' reserved mobile application, they can asset their disponibilty and can answer to a call: in a positive way, by accepting the request, or in a negative way by dismissing it. They will be advised about their duty by the system.

1.5 Goals

The application that will be developed is called myTaxiService and is both a mobile and a web application that allows users to easily call for a taxi or make a reservation. There is also a drivers' reserved mobile application that allows taxi drivers to simplify their job.

The application will allow visitor to register into the system.

It will permit to users to log in. After having logged in users can make

a quick call by providing their location or a reservation by providing departure date and time, departure location and arrival location. Logged users can also delete quick calls, within a minute, and reservations, by two hours before departure time. Right after a quick call users will receive a notification, whereas reservations' notifications will be provided ten minutes before departure time. Users can also modify their profile, view their rides' history, and check the last unfulfilled request. On the other hand the application will allow taxi drivers to log in. After having logged in taxi drivers can check their calls list, accept or reject a call, check the waiting queue and inform the system about their availability.

1.6 Definitions, Acronyms, Abbreviations

1.6.1 Definitions

- Quick call: the users can only specify the origin of the ride and is supposed that the taxi is request immediatly
- Reservation: the users can do this reservation at least two hours in advance of the meeting time by specifying the departure and arrival of the ride
- Taxi driver: a person whose job is to drive a taxi. It works for the governament of the city
- Drivers' reserved mobile application: an application that is given to taxi drivers from the taxi company. It allows the driver to asset their disponibilty and can answer to a call: in a positive way, by accepting the request, or in a negative way by dismissing it.
- Taxi identification code: each taxi has an unique identification code written on each back door that allows the user to easily find their own taxi.
- Reservation number: unique id that represent the user reservation.

- Driver's calls list: it contains all the drivers call up to a month before the date of the visualization. It shows the reservation number, the date, the departure location and time and the arrival location and time of the call.
- User's rides history: it lists all the user's reservations and quick call, ordered by date. It shows the reservation number, the taxi identification code, the price, the date, the departure location and time and the arrival location and time of the call.
- Taxi driver's next call: it is the call that the driver has accepted but has not accomplished. There can only be on next call at the time, so that a driver can not get more than one call at the time.
- User's last unfulfilled request: it is the last request that the user has made that has not been fulfilled. In other words it is a quick call or reservation in which the taxi driver has not picked up the user yet.
- Notifications: pop up messages shown to the user or to the driver in order to give them information about rides.

1.6.2 Acronyms

RASD: requirements analysis and specifications document

1.7 Reference Documents

- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.
<http://www.cs.tau.ac.il/~tyshbe/IEEE-830/IEEE-830-1993.pdf>
- <http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5841>
- Alloy documentation:
<http://alloy.mit.edu/alloy/>

- Specification Document: Assignments 1 and 2 (RASD and DD).pdf.
- Slides from Software Engineering 2 course of Politecnico di Milano, academic year 2015/2016.
- Examples from past years.

1.8 Document overview

The system to be implemented must have all characteristics defined in this document.

This document is essentially structured in four part:

- Section 1: Introduction
It provides a view of the main features and functionalities offered by the application.
- Section 2: Overall Description
It gives general information about the software that must be implemented with more focus on constraints and assumptions.
- Section 3: Specific Requirements
this part list requirements, typical scenarios and use cases. To give an easy way to understand all functionality of this software, this section is filled with UML diagrams.

Using more details, this document is divided into the following topics:

- Product Context: subsection that defines the aspects of cooperation between the system to be implemented and the existing systems.
- Product Functions: subsection that summarizes the main functions of the software system which will be developed.
- User Characteristics: subsection that describes the features and basic skills required for users of the system (e.g. Levels of education and skills).

- Constraints and Assumption: subsection that lists all the additional constraints that limit the design options available to developers and lists the assumptions made by developers in the preparation of this requirements document.

The Specific Requirements section is instead divided into the following subsections:

- External Interfaces Requirements: sub-section that defines the types of input / output system.
- Functional Requirements: subsection that discusses the basic behaviours at the software level for correct acceptance and adequate processing of user input.
- Non functional requirements: subsection that analyse in detail the aspects of service quality.
- Design Constraints: subsection that describes the design constraints of the system.

2 General description

The following section describes the general aspects that affect the system and its requirements.

2.1 Product context

The application we will release is a web application or mobile app which is not yet integrated with other existing system. There will be two different mobile apps, with different functionalities, one addressed to customers and the other to taxi drivers. The web application will not have any internal interface for administration but it will only be user based. The mobile application in the future will provide interaction with third-part system such as navigation systems in order to allow a notification management according to traffic conditions. The application will provide interfaces or API for integration with future projects, such as the share a taxi functionality.

Both the web and the mobile application provide interactions with a security system in order to preserve client's credit cards information.

2.1.1 System interface

The system must use Internet to offer the service. It also has to have access to the taxi's GPS in order to manage the taxi queues and to the payments database.

2.1.2 User interface

The users can use myTaxiService web application with the following browsers:

- Chrome
- Internet Explorer
- Mozilla Firefox

- Safari

The users can use myTaxiService mobile application with the following operative systems:

- iOS
- Android
- Windows phone

Taxi drivers are provided with a smartphone with myTaxiService already installed on.

2.1.3 Software interfaces

Data base management system:

Name: MySQL Community Server

Version: 5.6.21

Source: <http://dev.mysql.com/>

Java virtual machine:

Name: Java EE

Version: 7

Source: <http://www.oracle.com/technetwork/java/>

Application server:

Name: GlassFish Server

Version: 4.1 open source edition

Source: <https://glassfish.java.net/>

Operating System:

The software may be installed on any hardware with any operating system that supports the Java Virtual Machine, Database and application servers indicated above.

2.1.4 Operations

The Interaction between system and user is different according to the classification of the latter. For each user, the specification of permitted operations is described in the “2.2. Product Functions“.

2.2 Product functions

- Registration to a person into the system
- Login for a registered user
- Login for taxi driver
- Reservation of a taxi for a specified time providing departure time and location and arrival location

2.3 User Characteristics

The user of mytaxiService is a person who is able to access to the internet but no particular capacity is required to use our application. It is destined to people who uses to take taxi to moves in the city and would try an easier and faster method to do it. The only constraint is to have a credit card account.

2.4 Constraints and assumptions

2.4.1 Hardware limitations

To use our application the user need to have an Android or iOS smartphone with a internet connection for mobile application or a pc with internet connection for the web ones.

2.4.2 Interface to other applications

myTaxiService does not interact with any application at the moment, but in they future it may interact with applications such as Google Maps.

2.4.3 Requirement of high-level language

The development and management of the myTaxiService platform requires knowledge about J2E, HTML, CSS, JSF and MySQL.

2.4.4 Reliability requirements

In management conditions of a normal data stream is not required any constraint in order to maintain correct operation.

2.4.5 Security considerations

The development of myTaxiService's system must take in consideration security issues linked to the management of credit card data. In order to preserve them myTaxiService has to provide interactions with an existing security system.

2.4.6 Other considerations

Once finished myTaxiService should be:

- Easy to use: so that user doesn't have to spend a lot of time to make the requested operations and taxi drivers can easily access to important information in order to simplify their job.
- Have a nice look and feel: the design must be user-friendly and has to have a beautiful design in order to make a pleasant use.

2.4.7 Assumptions

- A user can not create more than one account.
- Non registered users can only register one account.
- A user can not do a reservation for the past.
- A user can not do a reservation more than two week before.
- A user can not do more than one reservation at the time.
- A user can not do a reservation or quick call if he does not have enough money on his bank account.
- A user can delete a reservation untill two hour from the reservation departure time or he will lose his money.
- If a user deletes a quick call or reservation he can do it. This means that the time constraint are respected.
- All the taxi drivers are already registered in myTaxiService's database by the government.
- If a taxi driver does not answer to a call within a minute the system behaves like he has rejected the call and his status is setted to not available.

- If a reservation is deleted the user will receive notifications about that reservation no more.
- If a user does not associate a bank account to his myTaxiService account he can not do a reservation or a quick call.
- A user can not associate more than one bank account to his myTaxiService.
- GPS positions are always 100% accurate.
- There is always at least one taxi driver that can solve a quick call.
- There is always at least one taxi driver that can solve a reservation.
- In a reservation the departure location is always different from the arrival location.
- Each user has a valid credit card. This means that it isn't expired and can be used for online transactions.
- When the user signs up or logs in and when the taxi driver logs in the fields that he has to fill are not empty
- When a user makes a quick call or a reservation the fields he has to fill are not empty
- There are no notifications for a ride that does not exist
- There is one and only one notification for each ride
- The number of notification for each user must be equal to the number of rides
- The number of notification for each driver must be equal to the rides plus the rides he has rejected

- There is only one username per driver
- There is only one email and credit card per user

2.4.8 Future possible implementation

- A future functionality implementation is to allow the user to track the taxi position through the use of GoogleMaps.
- Another possible implementation is to allow the user to enable the taxi sharing option, this means that he is ready to share his taxi with others people if it is possible, thus sharing the taxi cost of the ride.
- In order to protect the on-line transmission of data an HTTPS protocol could be used. However this last proposal should be delayed to a future implementation because the HTTPS protocol needs the purchase of a certificate signed by recognized certification authority.

3 Specific requirements

3.1 External interface requirements

3.1.1 User interfaces

- Mobile application

Unauthenticated user login page



Unauthenticated user sign up page

Hand-drawn illustration of a smartphone displaying a 'TAXI' app sign-up page. The screen has a yellow background and contains several input fields for user registration. At the top, there's a status bar with 'ABC' and '10:39 AM'. The app header shows a menu icon and the word 'TAXI' with a taxi icon. The form fields include Name, Surname, Email, Password, Phone Number, Credit Card Type (dropdown), Credit Card Number, CCV, and Exp. Date (with a calendar icon). At the bottom are 'Cancel' and 'Submit' buttons.

Authenticated user home page



Authenticated user menu



Authenticated user ride's notification



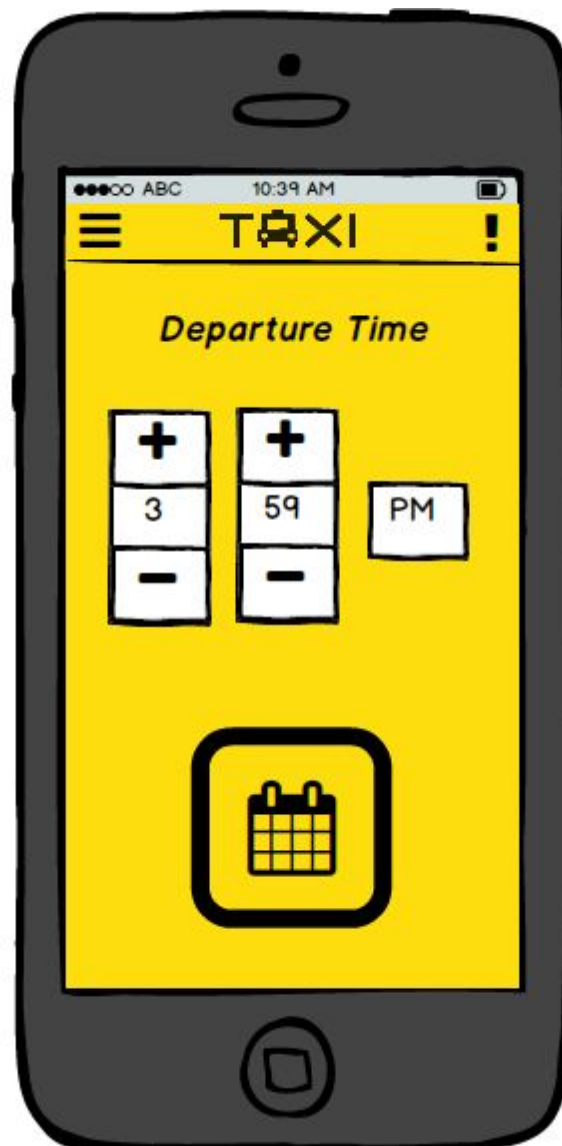
Authenticated user select reservation's destination location



Authenticated user select reservation's departure date



Authenticated user select reservation's departure time



Authenticated user modify personal information

The image shows a hand-drawn illustration of a smartphone screen. The screen displays a form for editing personal information within an app titled 'TAXI'. The form has a yellow background and is enclosed in a black border. At the top of the screen, there is a status bar with signal strength indicators, the text 'ABC', the time '10:39 AM', and a battery icon. Below the status bar is a yellow header bar with a hamburger menu icon on the left and the word 'TAXI' in the center. The form contains the following fields: 'Email', 'Password', 'Phone Number', 'Credit Card Type' (a dropdown menu with a downward arrow), 'Credit Card Number', 'CCV', and 'Exp. Date' (a date picker with a calendar icon). At the bottom of the form are two buttons: 'Cancel' and 'Save'. The smartphone itself is dark grey with a circular home button at the bottom.

- Web application

Unauthenticated user home page




The screenshot shows a web browser window titled "A Web Page" with the address bar displaying "https://www.mytaxiservice.com/LogIn.html". The page has a yellow background and a black "TAXI" logo with a car icon in the top left. In the top right, there are input fields for "UserName" and "Password", a "Login" button, a "Keep me logged in" checkbox, and a link "Forgot your Password?". The main content area is split into two sections. The left section features a large black icon of a person with a briefcase hailing a taxi. The right section is titled "Sign UP" and contains a form with the following fields: "Name", "Surname", "Email", "Password", "Phone Number", "Credit Card Type" (a dropdown menu), "Credit Card Number", "CCV", and "Exp. Date" (a date picker). A "Sign Up" button is at the bottom of the form.

Authenticated user home page

A Web Page




https://www.mytaxiservice.com/home.html


TAXI




 <Surname> <Name>  

Quick Call




Q Departue


A   

A 

A   

Q Arrival




A   


A 


Make a Quick Call


Reservation

Q Departue




A   


A 

gg/mm/aaaa 

hh:mm:ss 

Q Arrival

A   

A 

Make a Reservation

Request #123

Type: Call Type

Date: gg / mm / aaaa

Time: hh . mm . ss

Departure: DepartureAddress

Destination: Destination Address

Remaining time to delete:
hh : mm : ss

Delete

Type: Call Type

Date: gg / mm / aaaa

Authenticated user notification's information

A Web Page

https://www.mytaxiservice.com/home.html

TAXI

<Surname> <Name>

Notification Info

Quick Call

Departue

A

A

A

Arrival

A

A

Make a Quick Call

Reser

Departue

A

A

gg/mm/aaaa

hh:mm:ss

Arrival

A

A

Make a Reservation

t #123

aaa

Time: hh : mm : ss

Departure: DepartureAddress

Destination: Destination Address

Remaining time to delete:
hh : mm : ss

Delete

Type: Call Type

Date: gg / mm / aaaa

Authenticated user settings management

A Web Page

http://www.mytaxiservice.com/settings.html

TAXI

<Surname> <Name>

Settings

<Email> <Password>

<Phone Number>

<Credit Card Type> <Credit Card Number>

<CCV> Exp. Date <gg>/<mm>/<aaaa>

Save Edit

Request #123

Type: Call Type

Date: gg / mm / aaaa

Time: hh . mm . ss

Departure: DepartureAddress

Destination: Destination Address

Remaining time to delete:
hh : mm : ss

Delete

Type: Call Type

Date: gg / mm / aaaa

3.1.2 Taxi driver interfaces

Unauthenticated taxi driver's home page



Authenticated taxi driver's home page



Authenticated taxi driver notification's information



Authenticated taxi driver manage settings panel



3.2 Functional requirements

The system will be offering two kind of services: to the user an easy way to quick call or reserve a taxi, and to the taxi driver a simple way to manage calls.

The system must provide different functionality depending of the type of end user:

3.2.1 Unauthenticated user or visitor

Visitors can only register into the sistem. After the registration they became unauthenticated users.

Unauthenticated users can only sign in and thus log in to take advantage of all the services offered by the system.

3.2.2 Authenticated user

Authenticated users can make a quick call by providing the location where they want to be picked up, or make a reservation by providing departure date and time, departure location and arrival location.

They can also delete quick calls, but it has to be done within a minute. Reservations, instead, can be deleted by two hours before departure time. It is also possible to manage the last unfulfilled request, in order to check the taxi arrival time, the taxi identification code and the reservation number.

Authenticated users can also enter in notification panel that will show them the last 10 notifications about their rides.

Quick calls notifications are provided right after the taxi driver's confirmation, whereas reservations' notifications will be provided ten minutes before departure time.

Authenticated users can also have access to their profile in order to check information or modify it, for example updating credit card data or change password.

Another feature to which authenticated users can be admitted is the rides' history, where they can check the reservation number, the taxi identification code, the price, the date, the departure location and time and the arrival location and time of the rides they have made.

3.2.3 Unauthenticated taxi driver

Unauthenticated taxi drivers can only sign in and thus log in to take advantage of all the services offered by the system.

3.2.4 Authenticated taxi driver

Authenticated taxi drivers can check their calls list in order to manage salary and shifts, can inform the system about their availability or business and, when the system sends them a call notification, they can decide whether to take the call or not.

3.2.5 Scenarios

Unauthenticated User

- ***Code SC01***

Title : User registration

Description : Sebastiano is going to a family dinner and he is in a hurry when he discovers that the subway is blocked. He has heard some friends talk about a new application, called myTaxiService, and he decides to download and try it. In order to call a taxi he needs to register first. He fills the fields with his name, surname, email address, password and credit card data. A few minutes later he receives a confirmation email and he is ready to go.

- **Code SC02**

Title : Log in

Description : Luca is coming back home with his new girlfriend Cristina. He is registered at myTaxiService and decides to use it. He is unauthenticated and in order to utilize the application he has to log in by providing his password and email address. After the system has recognized Luca, he can call a cab.

Authenticated User

- **Code SC03**

Title : Make a reservation

Description : Elena and Niccolò are two friends that are going out on Saturday night. Since none of them wants to remain sober and drive they decide to reserve a taxi by using myTaxiService. Elena is already logged in and so decides to use the make a reservation functionality. She is using the mobile app through her smartphone. She taps on the “make a reservation” button, then she has to add the departure location, then the arrival location and in the end the date and time of the departure. Once she has set those parameters she can enjoy her night out with Niccolò.

- **Code SC04**

Title : Make a quick call

Description : Ilaria has to take a plane to go on vacation. Since she will be gone for two weeks and she doesn't want to pay much for the parking ticket, she decides to call a cab. She is already logged in and so decides to use the quick call functionality. She is using the web application via Safari, so she has to click on the quick call button, set the departure location and wait for the notification. Once set, she is ready for her vacation

- **Code SC05**

Title : Look at notifications

Description : Marco is planning to surprise his girlfriend for their anniversary. He has planned everything: he has called the restaurant, bought the flowers and reserved the taxi for her via myTaxiService. He need to tell his girlfriend the number of reservation and the taxi identification number that were provided him with the notification of the arrival time. He is already logged in his myTaxiService application and he only has to tap on the notification button in order to see all his notification. By selecting the last one he can provide his girlfriend all the notification she needs, and they can enjoy their first year togheter.

- **Code SC06**

Title : Modify profile

Description : Alessandro is a myTaxiService user. He has made several rides with this application and he really likes it. His credit card is expiring and he decides to go to the bank and change it. Now that he has a new credit card, he has to modify the information about it on his myTaxiService profile. He is already logged in so he has just to click on the modify profile button. Once he has upgraded the information and clicked on confirm he has finished and can now use again myTaxiService.

- **Code SC07**

Title : Visualize rides history

Description : Pietro is checking his bank account. He is a very precise person, so he needs to visualize his myTaxiService rides history in order to double check it. He is already logged in, so he just has to click on the my rides history button. He can now see how much he spent on myTaxiService.

- **Code SC08**

Title : Delete reservation

Description : Michela was invited to Giorgio's house to have dinner together. She had made a reservation through myTaxiService. Unfortunately the day of the meeting Giorgio isn't fine and decides to cancel the dinner. So Micela has to delete her reservation. She is already logged in the system so she has to enter in last unfulfilled call area and then delete it.

Unauthenticated taxi driver

- **Code SC09**

Title : Driver's log in

Description : Maurizio is a new myTaxiService driver. He was provided with a smartphone with myTaxiService drivers application on it, a username and a passcode. He opens for the first time the application and so he has to enter his username and passcode in order to use the system. After the system verification he is ready to go.

Authenticated taxi driver

- **Code SC10**

Title : Show call list

Description : Silvia is a taxi driver that has just received her earnings. She wants to check if the balance is correct and so she has to accede to hers call list. She is already logged in and so she only has to tap on the show call list button. Now she can check her wage.

- **Code SC11**

Title : Visualize next call details

Description : Antonio is a taxi driver but he is a bit careless so he doesn't remember the street number of his next call. He is already logged in so he just has to tap on the visualize next call details button and he can check all the information he wants.

- **Code SC12**

Title : Notifications

Description : Giuseppe is a taxi driver who has just accomplished a call. He is parking the car when he ears a myTaxiService notification sound. He opens the application and discovers that he has a call. He decides to take the call by tapping the "accept" button.

- **Code SC13**

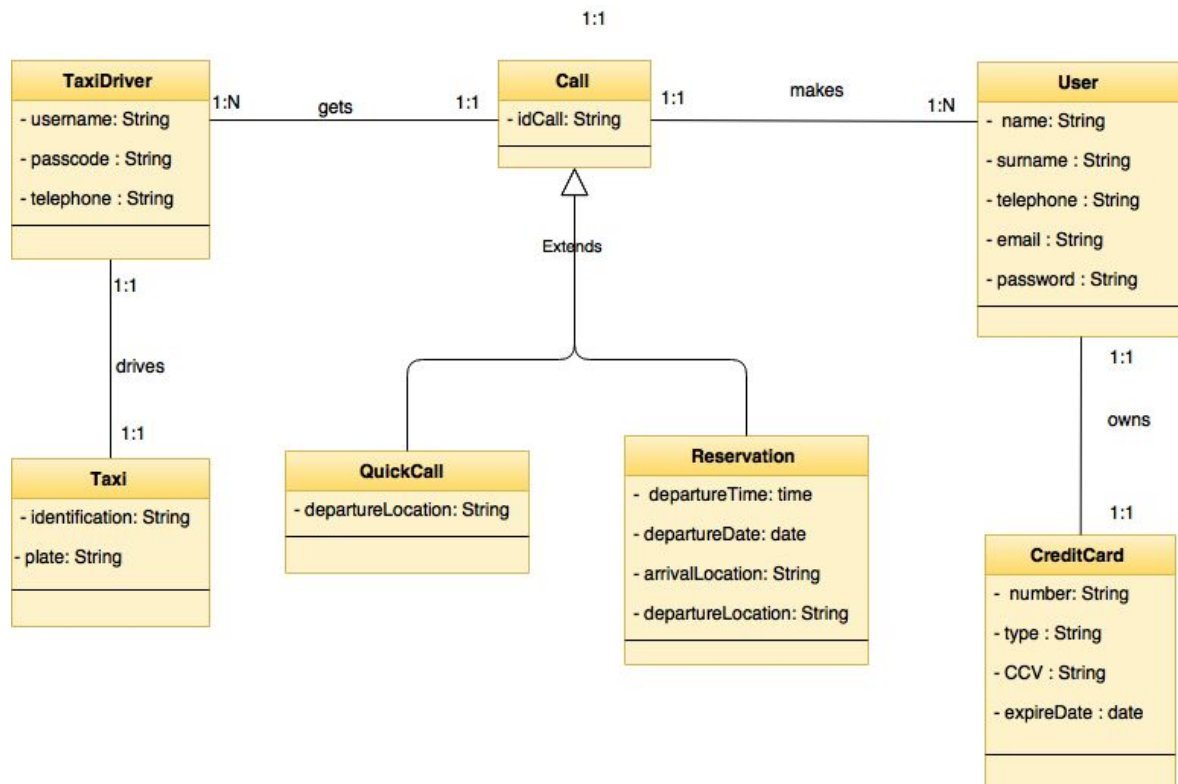
Title : Modify aviability

Description : Umberto is a taxi driver on service. He is waiting for a call when suddenly he receives a call from his child's school telling him that Mario, his little boy, is sick. So Umberto decides to change his aviability, by entering in his settings and switching the interruptor

to “not aviable” . Now that he is not callable by myTaxiService, he can run to his son.

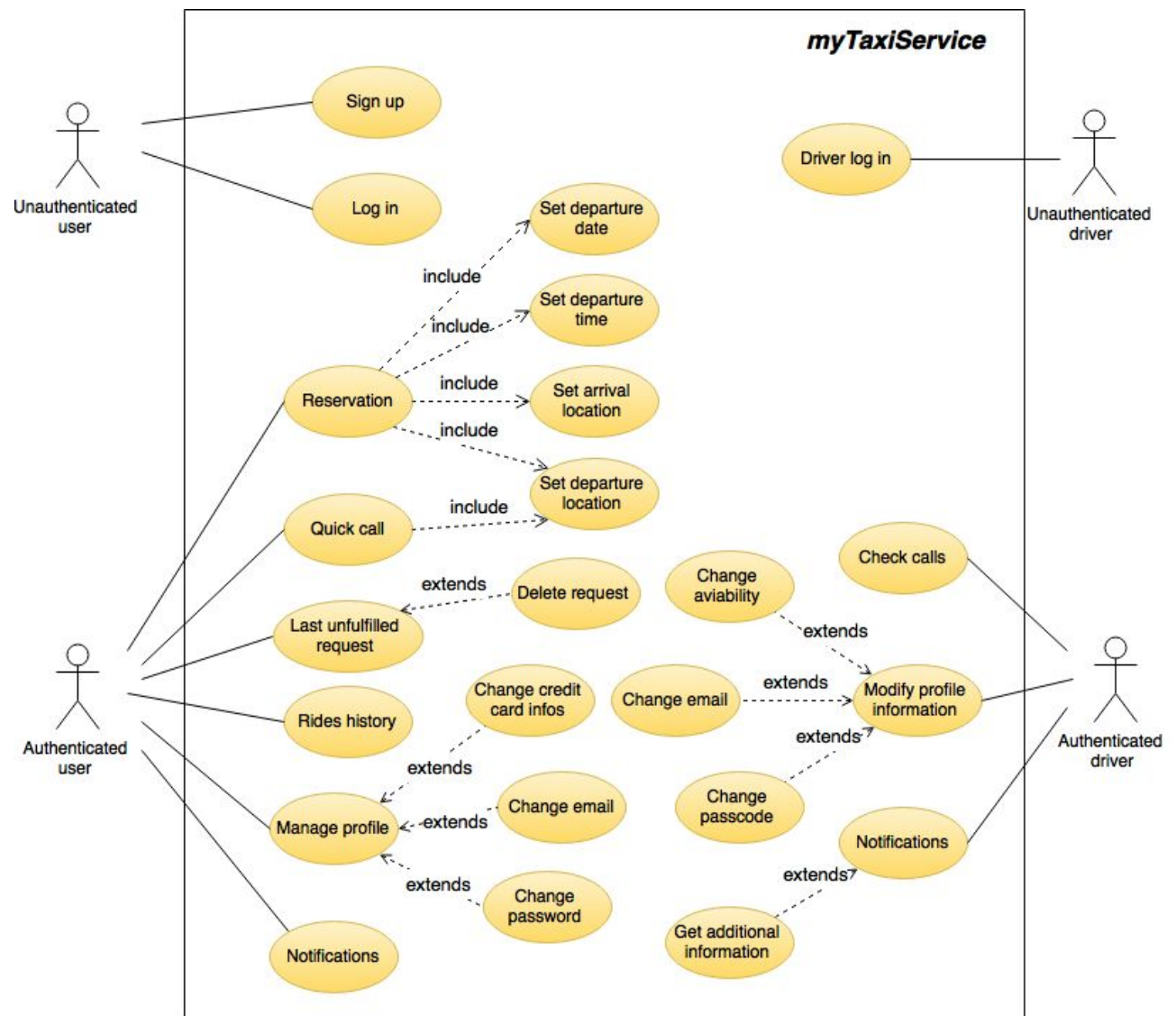
3.2.6 Analysis model

The analysis model represents the main system components.
The model is shown below with the UML Class Diagram.



3.2.7 Use case model

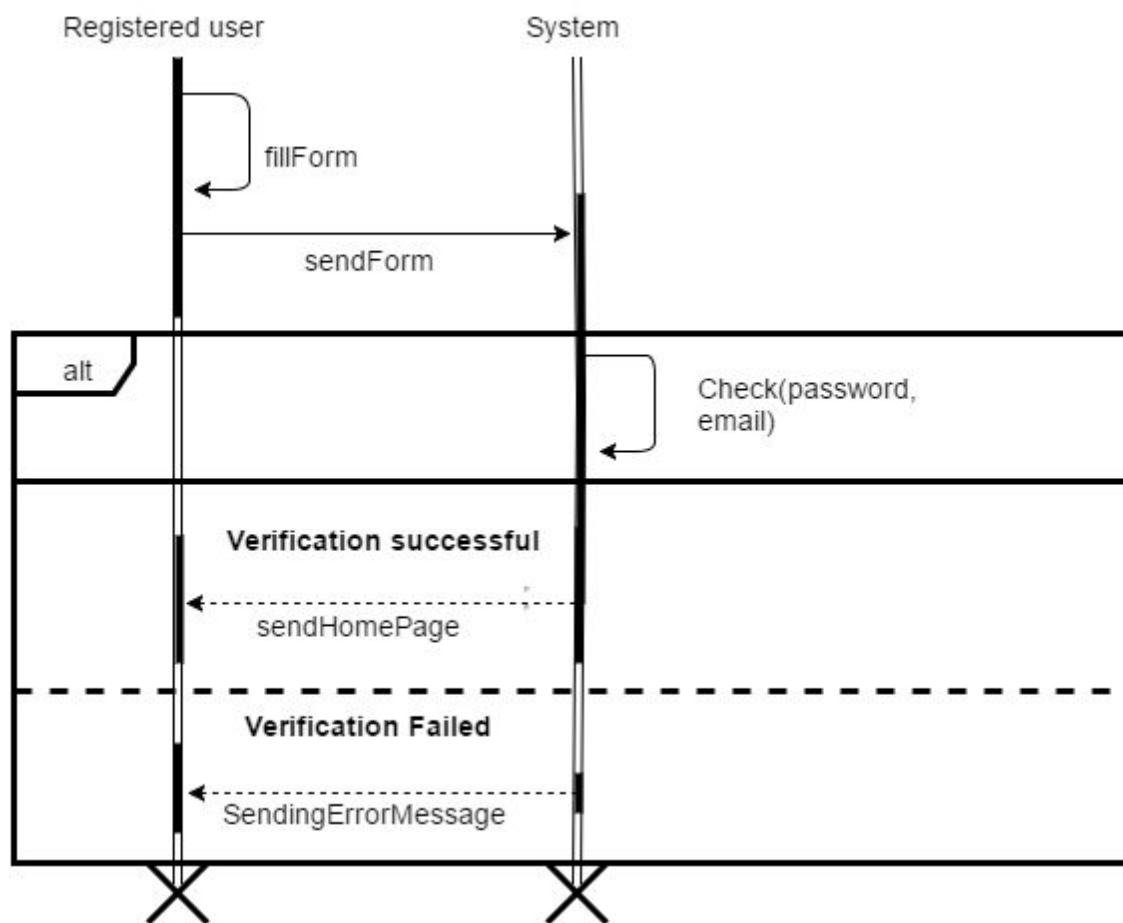
In this section is presented initially a general Use Case for the system, and then we'll analyse with more details the individual use case using UML Sequence Diagram.



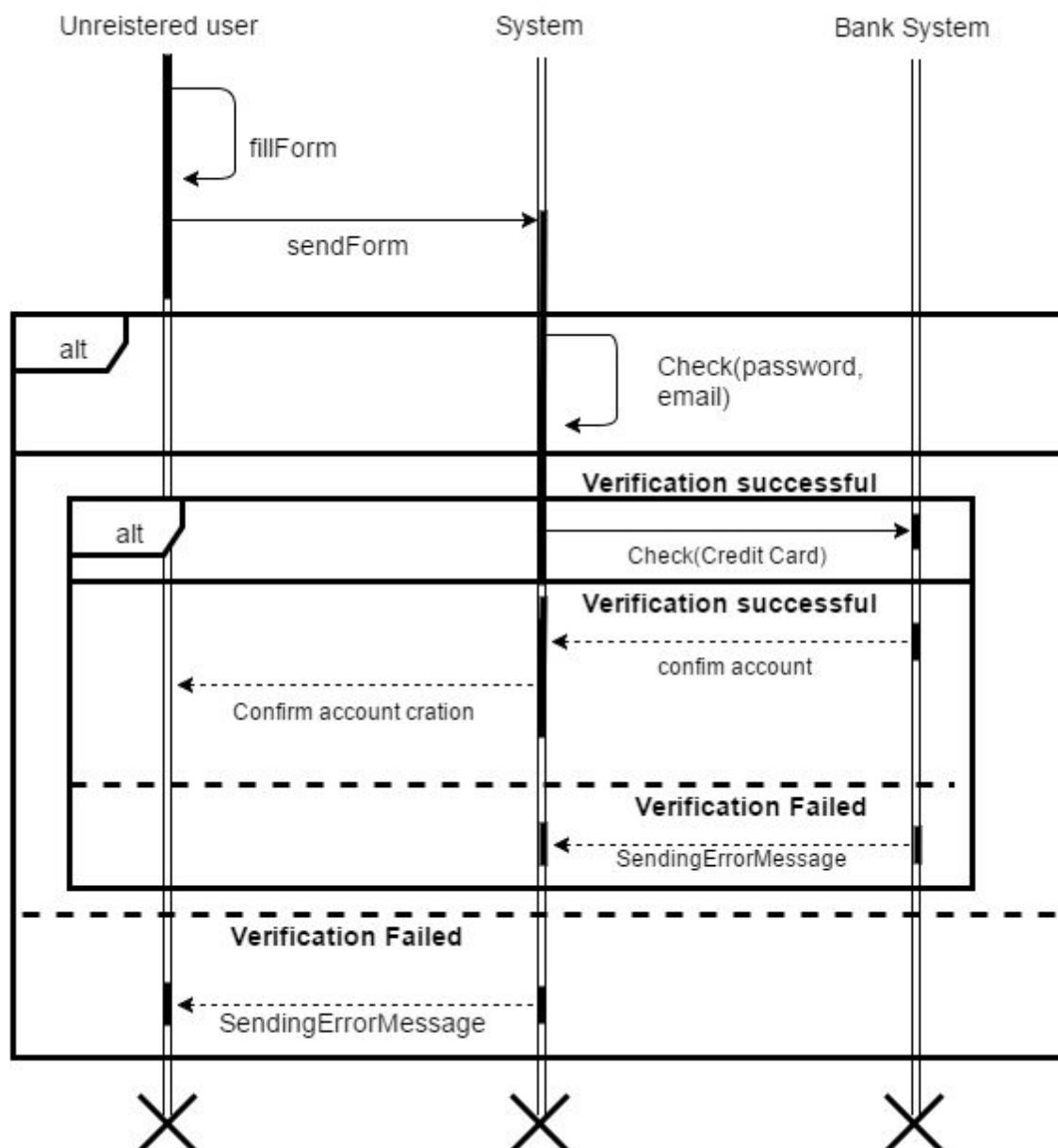
3.2.8 Use case description

Below are described in a detailed the main use cases with Sequence Diagrams.

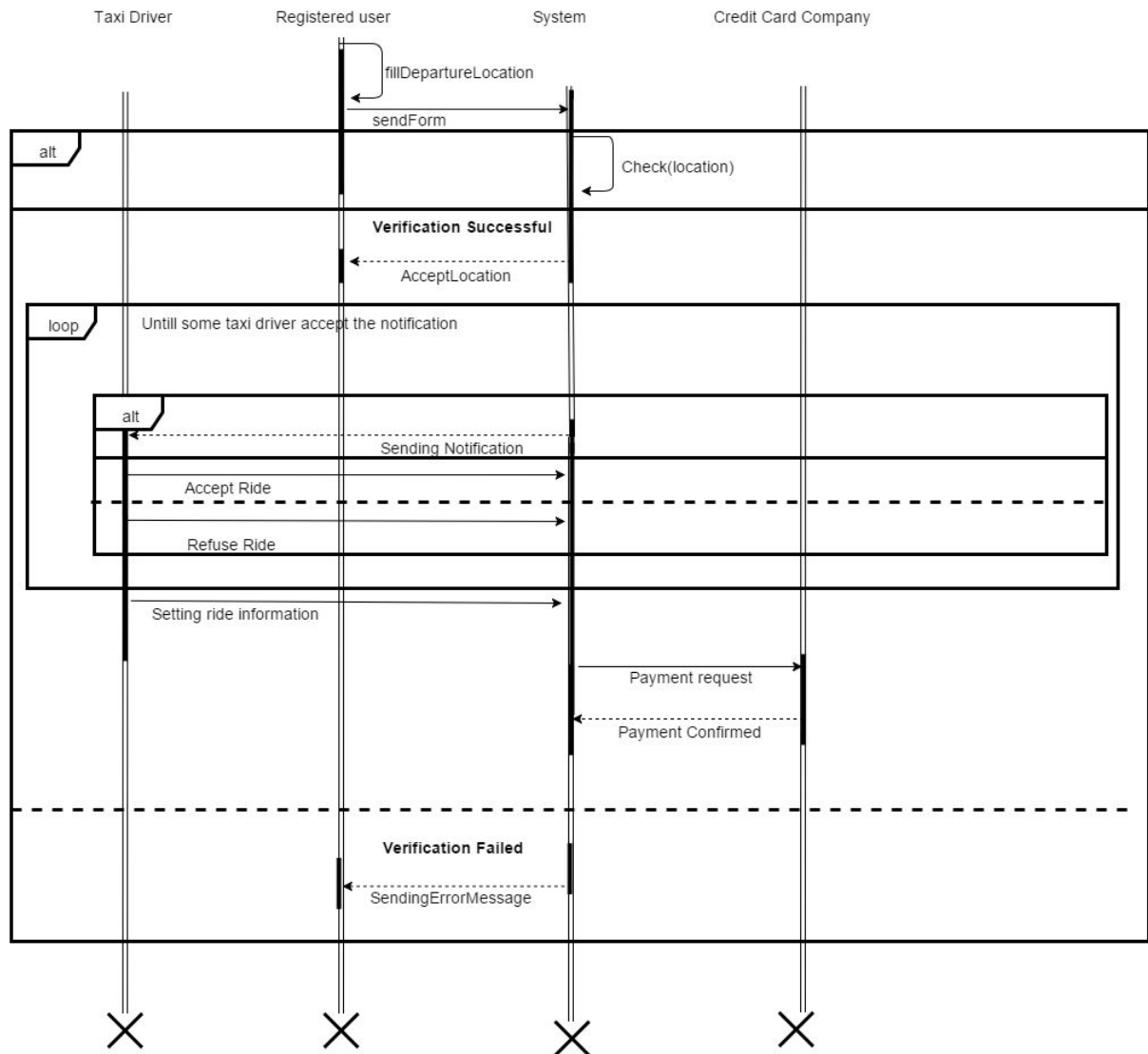
Name	Log In
Actors	Unauthenticated User
Entry condition	The user is correctly registered in the system and want to access.
Fow of event	<ul style="list-style-type: none">• The user open the Authentication page;• The system shows him that page;• The user enters his email and password in the input form provide;• The user click the button log in;• The system shows the user home page
Exit Conditions	The user is authenticated in the system.
Exceptions	The information inserted in the form are wrong, or he doesn't fill some mandatory fields. An error message is shown without showing to user own home page



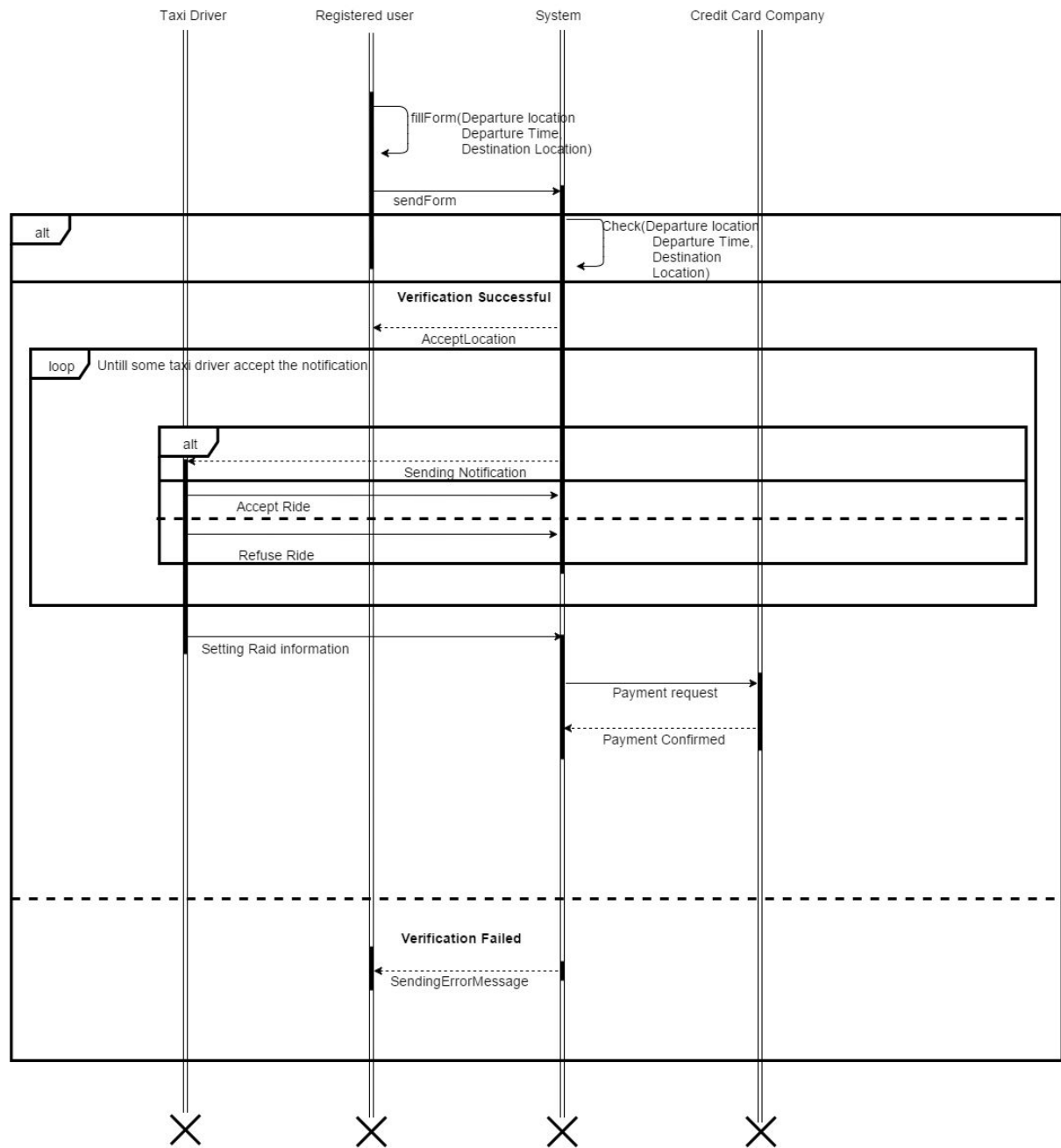
Name	Sign In
Actors	Unregistered User
Entry condition	The user whant to Sign in into the system
Fow of event	<ul style="list-style-type: none"> • The user open the Sign in page; • The system shows him that page; • The user enters his email, password, name, surname and credit card information in the input form provide; • The user click the button create; • the system control all the data; • The system shows the user successfully sign in page
Exit Conditions	The user is registered into the system.
Exceptions	The information inserted in the form are wrong, or he doesn't fill some mandatory fields. An error message is shown without showing to user own home page



Name	Quick call
Actors	Authenticated user
Entry conditions	The User has to be logged in and has to be in his home page. The user has to plan to do a quick call for a taxi ride.
Flow of Events	<ul style="list-style-type: none"> • The user insert the departue location; • the request is send to all the available taxi driver untill someone accept the request; • The user recive a notification about the succefully of the call; • After the ride the taxi driver send to the system theride information; • The ride is paid with the user's credit card;
Exit Conditions	The ride is complete and the payment is done.
Exceptions	The ride departure location does not exist. An error message is shown.

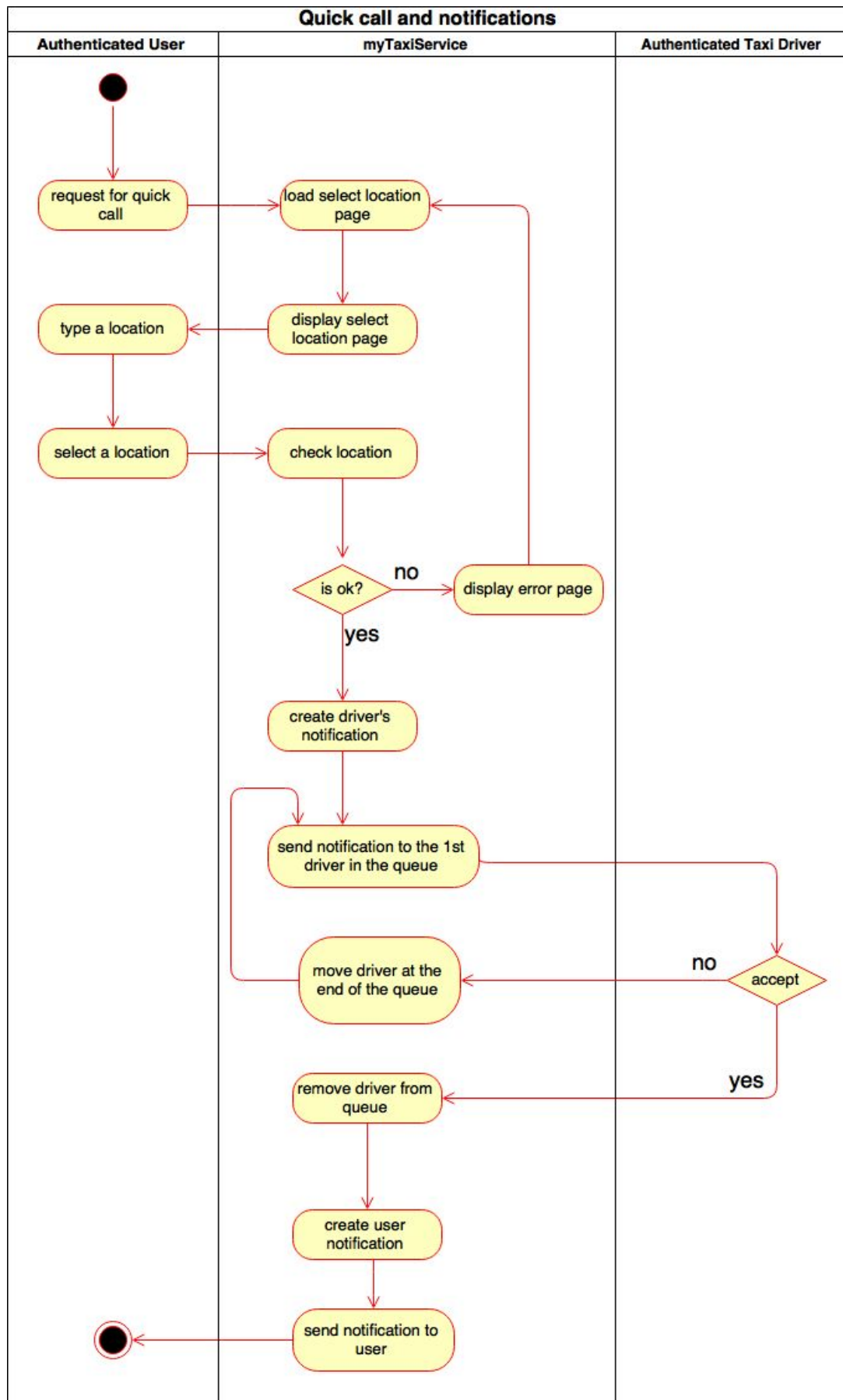


Name	Reservation
Actors	Authenticated user
Entry conditions	The User has to be logged in and has to be in his home page. The user has to plan to do a reservation for a taxi ride.
Flow of Events	<ul style="list-style-type: none"> • The user insert the time and the departure location; • The user insert the location of the destination; • the request is send to all the available taxi driver until someone accept the request; • The user receive a notification about the success of the call; • 10 minutes before the ride a notification arrive to taxi driver and user; • After the ride the taxi driver send to the system the ride information; • The ride is paid with the user's credit card;
Exit Conditions	The ride is complete and the payment is done.
Exceptions	The ride time or departure location is not valid or the destination location is not valid. An error message is shown.



3.2.9 Activity diagram

We analyze with the help of the Activity Diagram the case of a quick call event and the consequent taxi driver notification. This event is considered interesting to analyze in detail with this diagram since it covers both the user that the taxi driver. We must keep in mind the assumption about the drivers' queue: all the drivers in the queue are the one and only that are not involved in calls and that are available. After a call, if a driver is available, he is automatically added to the queue. When a driver accepts a call he is automatically removed from the queue. If a driver rejects a call he is moved at the end of the queue. If a driver does not answer a call the system takes it like a rejection.



3.3 Non functional requirements

3.3.1 Reliability

The system should be able to guarantee the service 24/24, 7/7. The server farm should allow every day maintenance without compromise the functionality. However the system doesn't cover a critical function, so brief unavailability could be acceptable.

3.3.2 Availability

There are checkpoints for every screen displayed so if a reservation or a quick call procedure is interrupted it will recovery from the last saved screen.

Data have to be always accessible, so the system has to provide always an access to them in normal condition. They also have to be duplicate in order to avoid data losses in case of system fault. For instance the use of RAID, mirroring and backup could present a valid solution.

3.3.3 Security

Both the log in parts, the one for the taxi drivers and the one for the users are crypted and protected by the couple username/email and passcode/password. All the log records are held in a database that is protected by a firewall and all the standar security measures for a secure database. Since the user payments are automatically executed after the ride, there is no need to restrict communications between some areas of the program or assign different function to different modules. However since the system allows the insertion of data, such data should be filtered in order to avoid vicious or unwanted modification in the database (e.g. SQL injection). At last to protect the on-line transmission of data an HTTPS protocol could be used. However this last proposal should be delayed to a future implementation because the HTTPS protocol needs the

purchase of a certificate signed by recognized certification authority.

3.3.4 Maintainability

The software is easy to maintain it implements no particular structures, the code is easy to read, not complex, commented and modular and so the maintenance can be easily done.

3.3.5 Portability

The software can be used with a large number of devices and browsers, so there are not problems with portability.

In particular the percentage of code that is host dependent is 0% and so is the percentage of component with host dependent code. No particular operating system is required.

3.4 Design constraints

The system must be developed with the technology JEE.

4 Appendix

4.1 Alloy

In this paragraph we try to understand if our class diagram can be consistent. In order to test the consistency of the class diagram we used Alloy.

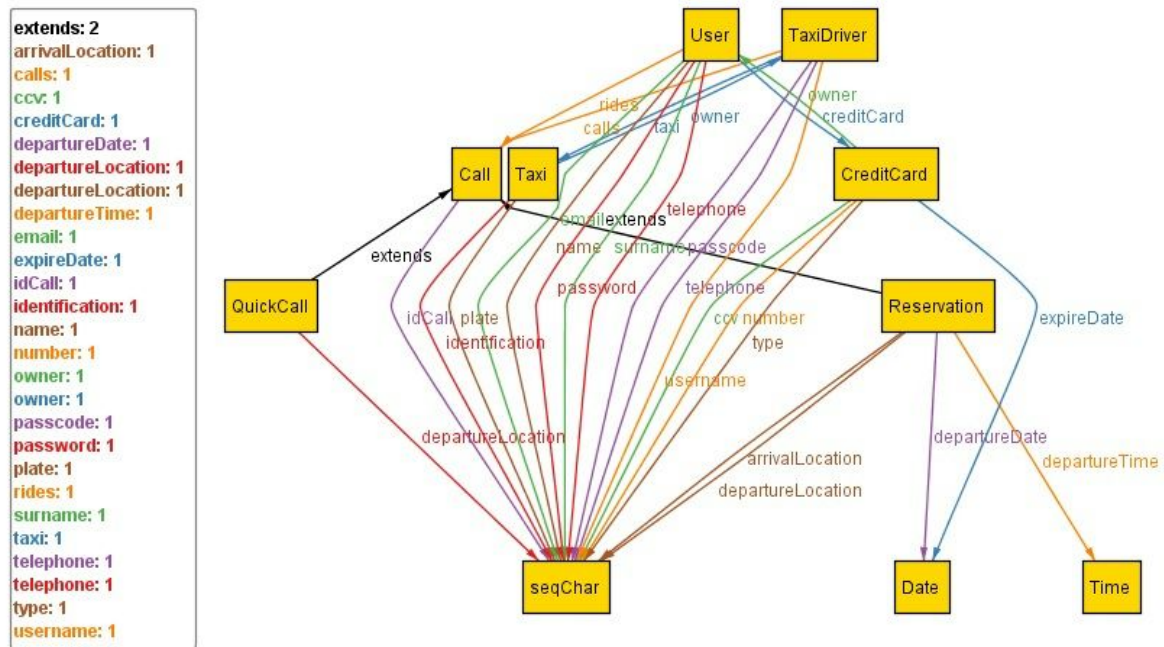
To define and understand the requirements of the system we have chosen to produce models using the specification language of Alloy. The problem has been split as follows:

- The entities of interest have been defined through “signatures”.
- The constraints of the system have been defined by “facts”.
- The verification of such constraints was made by “assertions”.
- The modelling of dynamic operations was made through “predicates”

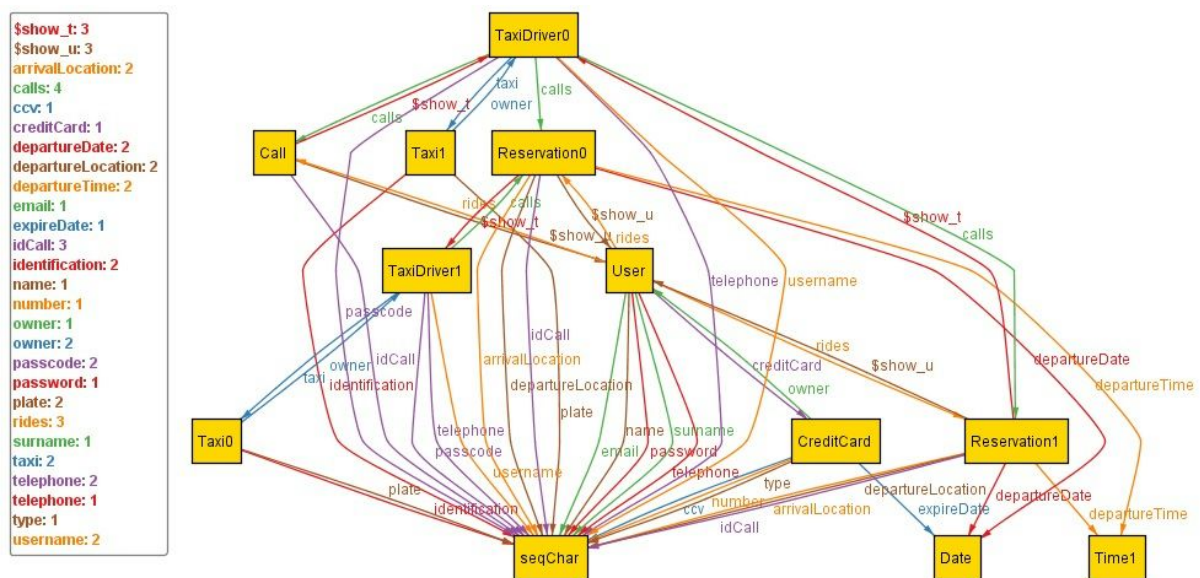
4.1.1 Diagrams

The following image shows the relation between the class of our class diagram. In particular we can see dependencies between classes and the attributes of each class. For example the class QuickCall and the class Reservation extend the class Call.

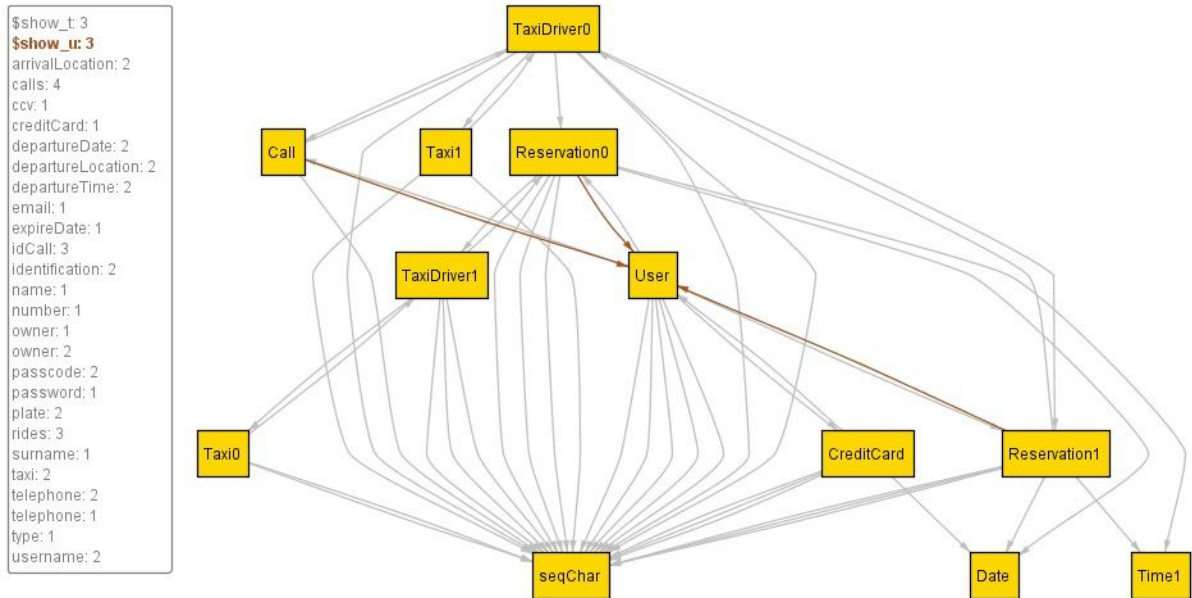
The class `seqChar` is used to represent a string.



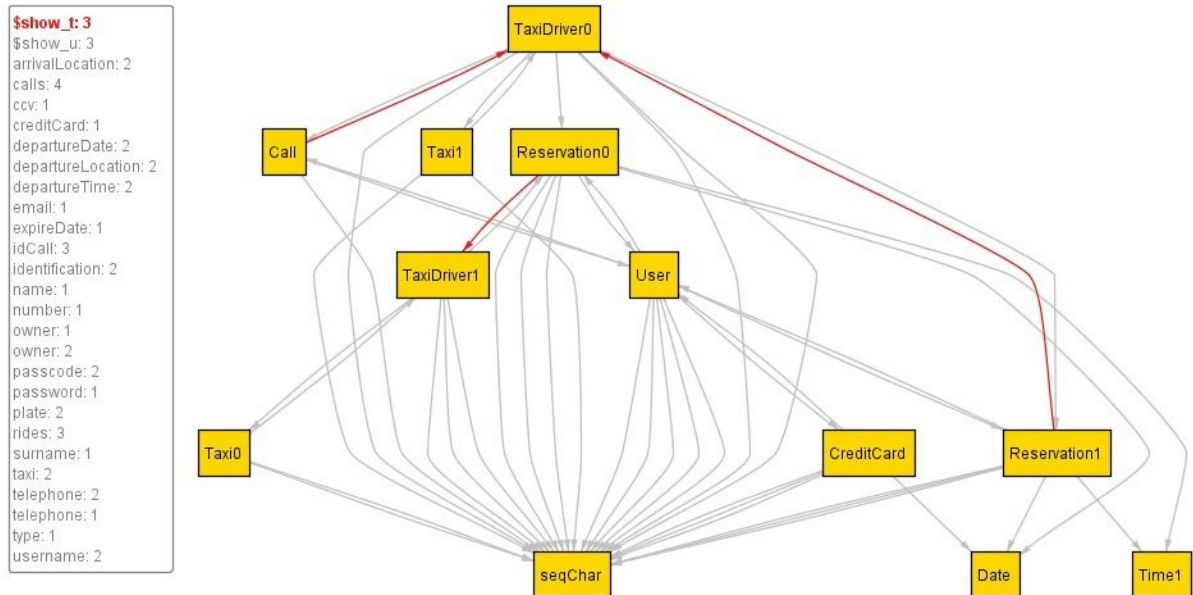
The next picture shows in a more specific way the representations of our model in order to better understand its characteristics. In it is represented one of the possible worlds consistent generated by the tool according to the requirements and specifications provided. The use of assertions has allowed detailing more precisely the set of requirements and constraints that define the system. Adding constraints to restrict the erroneous behavior has allowed a proper definition of the operations.



This diagrams put in result the connection between calls and user: each call must have one and only one user (the share a ride option will be implemented in the future). Although each user can make more calls.

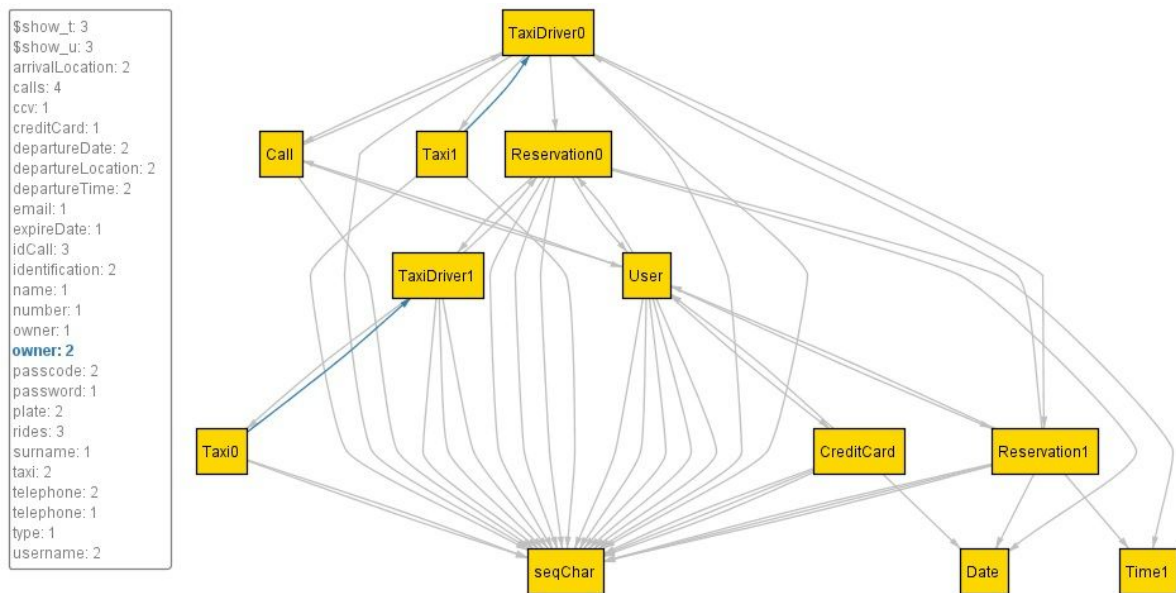


This diagram shows the connection between call and driver:
each call must be executed by one and only one driver, but a
driver can make more calls.

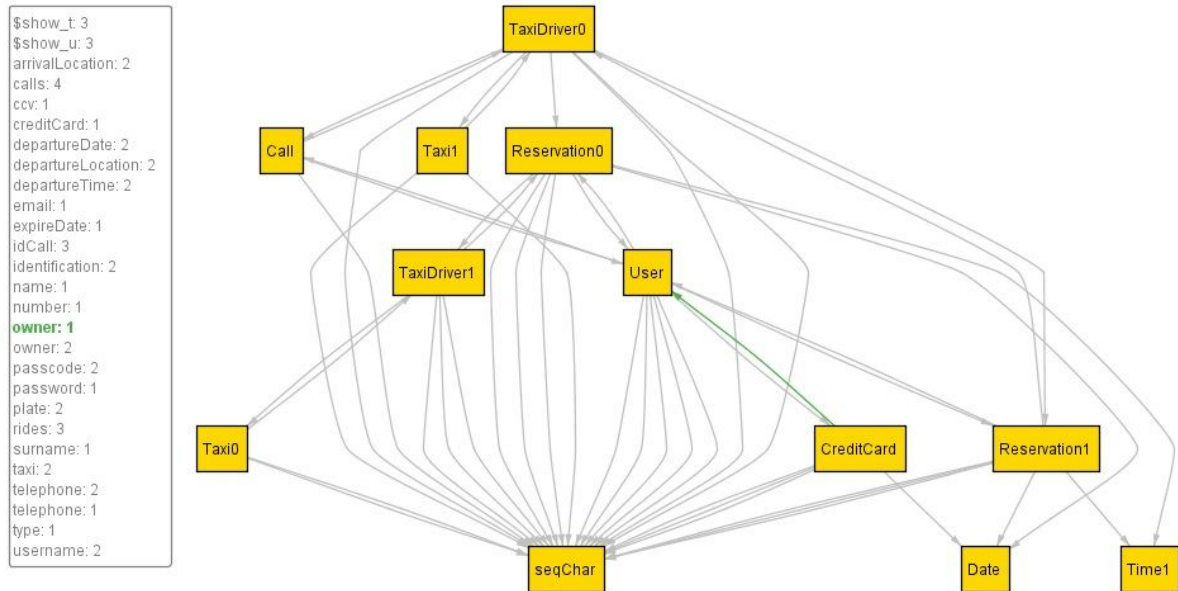


In this diagram is shown how each taxi must be associated to a taxi driver. Each taxi must have an owner, and the owner is unique.

We can see, similarly, that a taxi driver owns only one taxi.



The last picture presents us the connection between user and credit card. Each credit card has an owner and that owner is an unique user. Similarly, each user is connected to only one credit card at the time.



4.1.2 Code

Below is presented the code that was used in order to get the alloy diagrams showed in 4.1.1 .

```
module myTaxiService
  /** Class declaration */
  //SIGNATURES
  sig Date {}
  sig Time {}
  sig seqChar {}
  sig User {
    name : one seqChar,
    surname: one seqChar,
    telephone: one seqChar,
    email: one seqChar,
    password: one seqChar,
    creditCard: one CreditCard,
    rides: set Call
  }
  sig CreditCard {
    owner: one User,
    number: one seqChar,
    type: one seqChar,
    ccv : one seqChar,
    expireDate: one Date
  }
```

```
sig Taxi {  
    owner: one TaxiDriver,  
    identification: one seqChar,  
    plate: one seqChar  
}  
  
sig TaxiDriver {  
    username: one seqChar,  
    passcode: one seqChar,  
    telephone: one seqChar,  
    taxi: one Taxi,  
    calls: set Call  
}  
  
sig Call {  
    idCall: one seqChar  
}  
  
sig QuickCall extends Call {  
    departureLocation: one seqChar  
}  
  
sig Reservation extends Call {  
    departureTime: one Time,  
    departureDate: one Date,  
    departureLocation: one seqChar,  
    arrivalLocation: one seqChar  
}
```



```

/** Definition of the constraints */
//FACTS

// each user has only one credit card and each credit card is of a user

fact creditCardProperty {
    all u : User | ( all c : CreditCard | u.creditCard = c <=> c.owner = u )
}

// each driver has only one taxi and each taxi is of a driver
fact taxiProperty {
    all d : TaxiDriver | ( all t : Taxi | d.taxi = t <=> t.owner = d )
}

// each call in taxi driver's calls must be in a user rides

fact callConsistency {
    all c : Call | some u : User | some t : TaxiDriver |
        c in u.rides && c in t.calls
}

/** Verify the model */
//ASSERT

//Each user has only one creditcard and that credit card is of that user

assert onlyOneOwnerCreditCard{
    all u : User |
        ( all c : CreditCard |
            ( u.creditCard = c <=> c.owner = u ) )
}
check onlyOneOwnerCreditCard

//Each taxi driver has only one taxi and that taxi is of that taxi driver

assert onlyOneOwnerTaxi{
    all d : TaxiDriver |
        ( all t : Taxi |
            ( d.taxi = t <=> t.owner = d ) )
}
check onlyOneOwnerTaxi

//Each call exists only if it is in user's rides and in driver's calls

assert callExistance {
    all c : Call | some u : User | some t : TaxiDriver |
        c in u.rides && c in t.calls
}
check callExistance

```

```
//PREDICATES

//add a ride to a user
pred addRideUser [u: User, r: Call] {
    u.rides = u.rides+r
}

//add a call to a taxi driver
pred addCallTaxi [ t: TaxiDriver, c: Call] {
    t.calls = t.calls + c
}

//delete a call
pred removeCall[c:Call, u:User] {
    u.rides = u.rides - c
}

pred show {}

run show
run addCallTaxi
run addRideUser
run removeCall
```

4.2 Tools

The tools we used to create this RASD document are:

- Google documents: to redact and write this document;
<https://docs.google.com/>
- Balsamiq Mockups: to create the icons and mock Up;
<http://balsamiq.com/products/mockups/>
- StarUML: to create Use Cases Diagrams and Class Diagrams;
<http://staruml.io/>
- Visio 2013: to create Sequence Diagram and State Charts;
<https://products.office.com/en/Visio/visio-pro-for-office-365-online-diagram-software>
- Alloy 4.2: to prove the consistency of our model;
<http://alloy.mit.edu/alloy/>

4.3 Working time

Group total amount : 96

- Antenucci Sebastiano 790021

Total amount : 48

Date	Worked hours
22nd october 2015	4
23rd October 2015	5
24th October 2015	3
25th October 2015	4
26th October 2015	3
27th October 2015	2
28th October 2015	2
29th October 2015	5
30th October 2015	3
31st October 2015	2
1st November 2015	2
2nd November 2015	5
3rd November 2015	5
4th November 2015	3

- Buonagurio Ilaria 852641

Total amount : 48

Date	Worked hours
22nd october 2015	4
23rd October 2015	5
24th October 2015	6
25th October 2015	6
26th October 2015	3
27th October 2015	2
28th October 2015	2
29th October 2015	6
30th October 2015	3
31st October 2015	2
1st November 2015	2
2nd November 2015	2
3rd November 2015	2
4th November 2015	3