Requirements Analysis and Specification Document

SmartCityAdvisor

Navid Heidari (798726) Hamidreza Hanafi (841408)

Sunday 26th June, 2016
version 2.0

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Purpose

This is the Requirements Analysis and Specification Document to be used in the design of the software system called *SmartCityAdvisor*. In this document we describe the specifications and constraints that the software we are to implement must have. The intended audience of this paper is:

- The project manager
- The client which in this case is the Government of the city
- The designers and developers of the application
- The testing team
- The end user

This document has contractual value.

1.2 Scope

Here we summarize the main scope of the application.
The client is the *government of Milano*.
The city of Milano has installed in its territory a number of sensors to acquire information about the following elements:

- Level of CO_2 in the air.
- Cars that enter in the city center.
- Availability of parking places in all areas in the city center.

So we received the request to design and implement this application, called *SmartCityAdvisor* which has basically four great objectives:

- limit the traffic that enters in the city center in special cases
- help citizens find best emergency room

- manage CO_2 level in the air
- find empty parking places for citizens

1.2.1 Users and interfaces of the application

The system is designed to interact with two kind of users:

- The citizens
- The control center

For each category of users we must provide an appropriate interface and the client requests for the interfaces are:

- A web application or a mobile application for control center
- A web application or a mobile application for citizens

1.2.2 High level behavior

If the level of CO_2 is too high or in case of any special situation managed by the control center (e.g., the arrival of some VIP, an accident,), limit the traffic that enters in the city center by diverting it through paths that avoid the center. This is done both by controlling the traffic lights and by alerting the citizens through the app and through some large displays that are installed at the main entrances of the city.
When a citizen signals through the app that he/she has to go to an emergency room, provide suggestions on which hospital to choose based on: i) the problem of the citizen and the specializations available in the hospitals, ii) the status of queues in the various emergency rooms, iii) the location of the citizen and iv) the situation of traffic.
If the level of CO_2 is too high, it should send alert to control center to manange city energy consumption to reduce CO_2 level in the air.
Help citizens to find empty parking places with respect to their location in the city center.

1.3 References

- IEEE Std 830-1998: *IEEE Recommended Practice for Software Requirements Specifications*
- Project document

1.4 Overview

In the next sessions of this document we will discuss about:

1. (Chapter 2) **Overall Description**
2. (Chapter 3) **Specific Requirements**

Chapter 2

Overall Description

2.1 Product perspective

SmartCityAdvisor will get the data of CO_2, car entry, hospital queues and parking slots from other systems, Also the traffic lights and LCD s are going to be managed by appropriate web services, So they are not going to be implemented.
SmartCityAdvisor will be implemented in a three-tier architecture. Server-side, on the application layer there will be the Traffic Light Manager (will take care of managing car entries in the city center) and the Notification System (will allow citizens and control center to get various advises upon their request or automatically). A dedicated server will host the database in which the system will store all the data.
Citizens will be able to access the system after a registration and a following login from both a web application and a mobile application (available for free for the three major mobile operating systems, Android, IOS, Windows Phone from their markets).
Control Center instead, will use a different version of the application that will not be available on the markets but will be provided by the government.

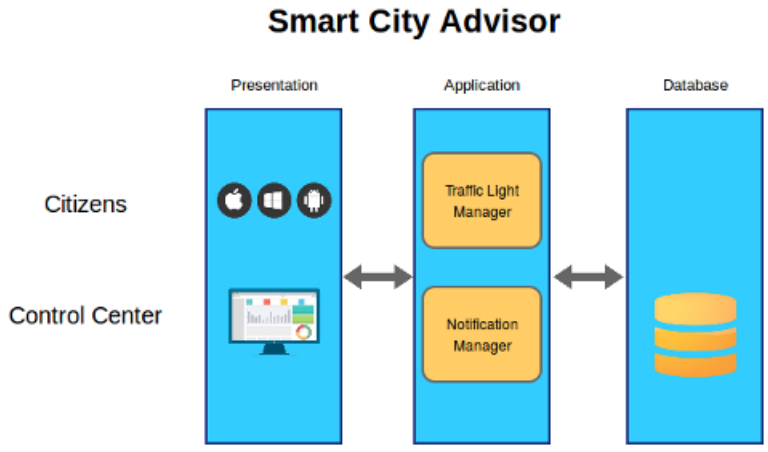


Figure 2.1: Schematic representation of *SmartCityAdvisor*

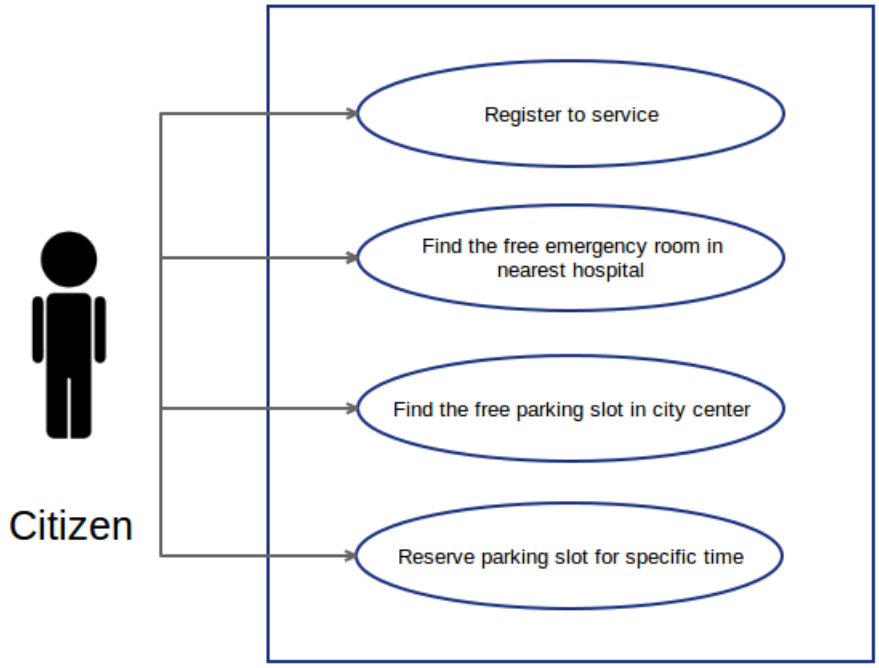
2.2 Product functions

The system offers the following functions with respect to the different users (Citizens and Control Center).

2.2.1 Functions for Citizens

A Citizen can:

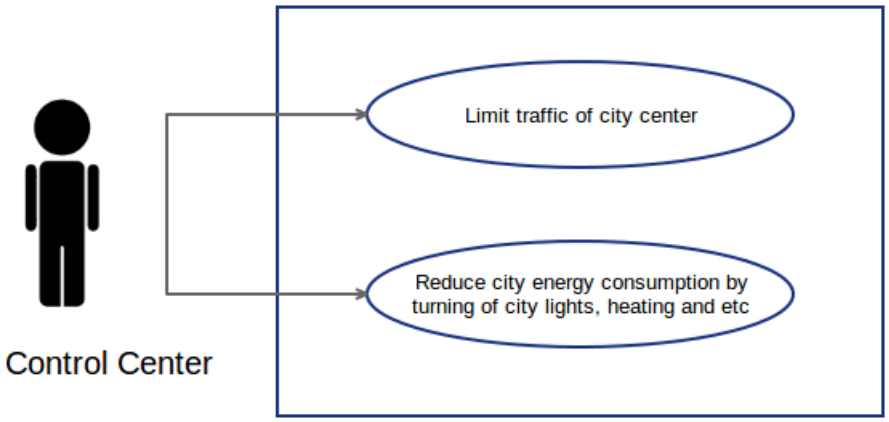
- Register to the service
- Find the free emergency room in nearest hospital
- Find the free parking slot in city center
- Reserve parking slot for specific time



2.2.2 Functions for Control Center

Control Center can:

- Limit traffic of city center
- Reduce city energy consumption by turning of city lights, heating and etc



2.3 User characteristics

The application addresses two specific kinds of users: *Citizens* and *Control Center*

2.3.1 User class 1: *Citizens*

A Citizen is a user with these characteristics:

- He owns a smart phone
- He is able to install a mobile application on his smart phone
- He owns/has access to a computer
- He has access to the Internet
- He is able to use a web browser

We can consider a citizen every person who wants to use the smart city adviser of the Milano city. They can be citizens of the city, tourists or simply visitors.

2.3.2 User class 2: *Control center*

Control center is a user with these characteristics:

- He is a employee of a government
- The government has registered the control center to the system
- He has access to a smart phone or a computer with web browser (provided by the enterprise)

Control center needs to work 24 hours a day and should monitor the city for the air pollution and emergency situations.

2.4 Constraints

2.4.1 Regulatory Polices

The software behavior must respect all the prescription of the local law about:

- Security and Integrity of the users data (both Citizens and Control center)

2.4.2 Hardware limitations

The system must be interfaced through two kind of mobile applications (one for the Citizen and one for the Control center) and a web application.

Mobile applications

Both applications must be developed for the three major mobile Operating Systems (Android, iOS, Windows Phone). The citizens version must be released in the three specific application market for free. The Control center version, instead, will only be available to the government. The applications have to request the minimum amount of authorizations to the smart phone Operating System and of course they must not damage/modify unrelated data stored in the device.

Web application

The web application must be supported by all the most famous web browsers: in particular Google Chrome, Safari, Mozilla Firefox and Internet Explorer.

2.4.3 Parallel Operation

The system must be able to deal with multiple contemporary requests coming from different Citizens.
The system must provide the right grade of parallelism.

2.4.4 Criticality

The system does present a critical application. The Citizens may use it to find nearest free emergency room.

2.5 Goals of the application

From the analysis of client's needs for the software to be we can derive the following list of goals.

2.5.1 Goals Citizen side

A Citizen:

G.P.1) must be able to register to the service

G.P.2) must be able to login to the service

G.P.3) can access *SmartCityAdvisor* either from the web application or the mobile application

G.P.4) must be able to search for emergency room close to his location with specific specialization

G.P.5) must be able to see empty parking spot in city center

G.P.6) must be able to reserve an empty parking spot in city center at a specific time, date and duration

G.P.7) must be able to cancel a reservation that he made

G.P.8) must receive, if an empty parking spot successfully reserved

G.P.9) if he has reserved an parking spot correctly for a certain time, he must be notified before that time with a notification saying the time of the start and it identification number, or, in case of problems, saying the nature of the problem.

2.5.2 Goals Control Center side

Control Center:

G.T.1) must receive a notification if the CO^2 emission is high or if there is special condition in the city center

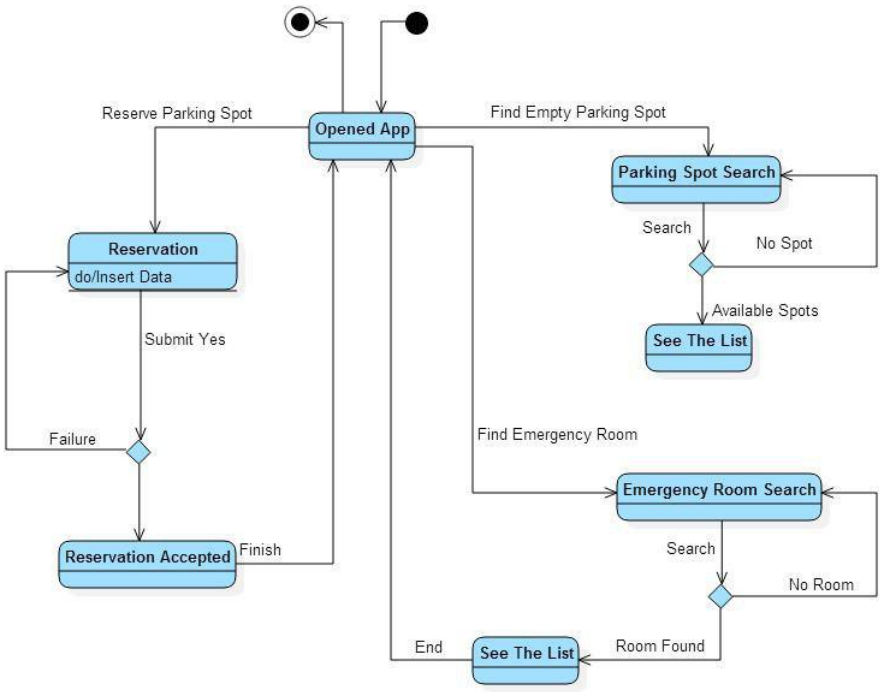
G.T.2) must control CO^2 emissions by controlling city energy use

G.T.3) must limit city center traffic in special conditions

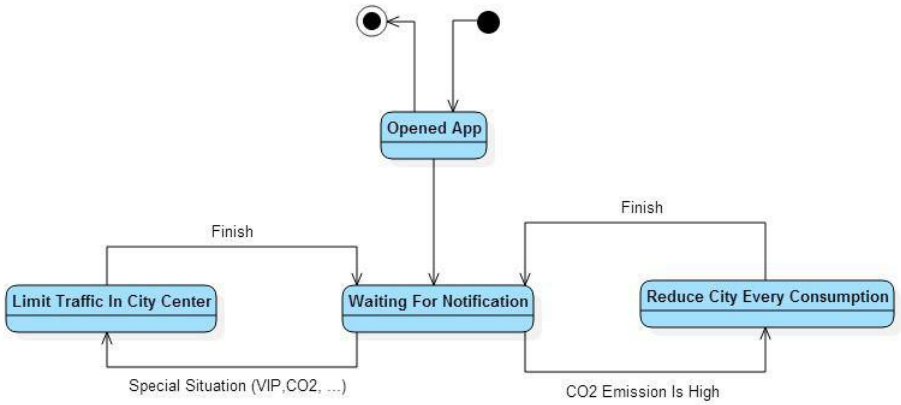
2.6 Derived behavior of the actors

From the domain analysis, the assumptions and the basic requirements for the application we can derive a model of behavior of the agents (Citizens and Control Center) when they use the future application

2.6.1 Citizen behavior



2.6.2 Control Center behavior



2.7 Assumptions and dependencies

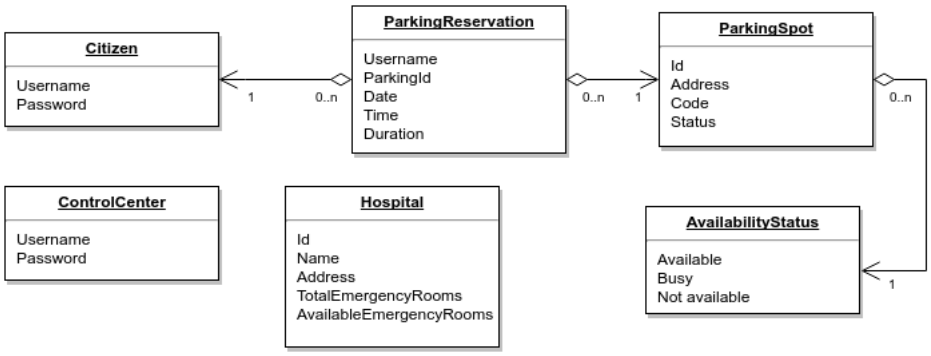
These are our assumptions about the problem domain:

- Each smart phone used by the Citizens has a GPS system installed and the application has the privileges to access it

- Once a Citizen made a reservation of a parking spot and it has been accepted and she/he's going to use it for the selected duration
- The Citizens need to register to the system (username + password) in order to use it
- Citizens can't access the mobile application used by Control Center

2.8 Entities involved

Here we summarize the entities involved in the system to develop



2.9 Future Possible Implementations

- The system could also manage the payment of the ride through an in-app interface (for parking reservation)
- The system will have the possibility to handle the cancellation of the reservation after its submission to the system.

Chapter 3

Specific Requirements

3.1 Scenarios

3.1.1 Registration to the service

Scenario

Bob has discovered the new service offered by the city called *SmartCityAdvisor* and he wants to discover how it works. He goes to the web application site and starts the registration phase. He insert the username and the password he wants and submit the request of registration. The system replies saying that the username selected is already used and that Bob has to insert another username. Bob insert another username and this time the procedure succeed. Bob is now correctly registered to *SmartCityAdvisor*.

Use case

Use case	**Register**
Actors	Citizen
Goals	A citizen must be able to register to the service
Enter condition	None
Event flow	1. The citizen goes to the web application page 2. The citizen clicks on the sign up button 3. The citizen fills the form with username and password desired 4. The citizen clicks the submit button 5. If the username is already present in the system or there are missing data (a) Notify the Citizen of the error (b) Go back to Event flow 3 6. The system retrieves the data and stores them 7. The system notifies that the registration has been correctly done
Exceptions	If the user, during the insertion of the registration data (username + password) decides to abort the procedure by clicking on the proper button 1. The system notifies the citizen of the loss of the inserted data 2. The system abort the procedure
Exit condition	The citizen is correctly registered to the service

Table 3.1: Use case: Register

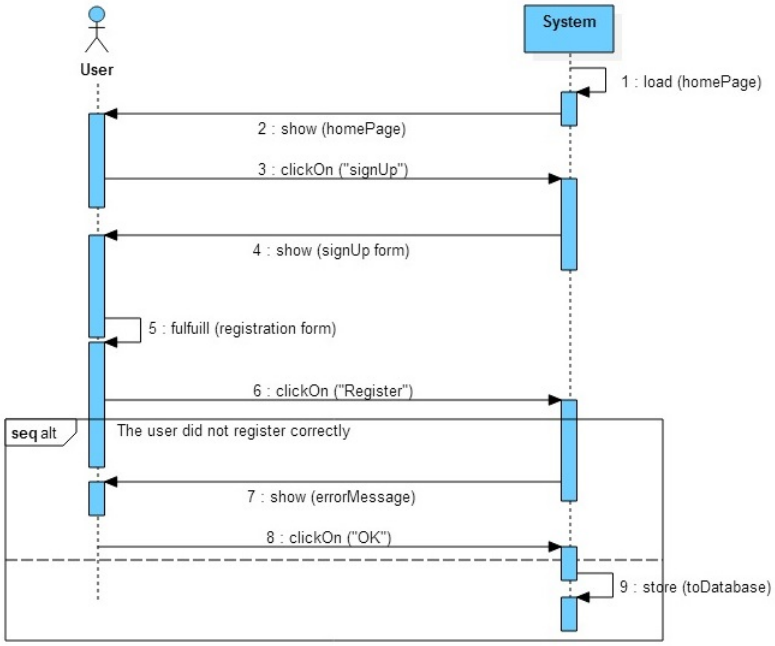


Figure 3.1: Registration to the service

3.1.2 Citizen emergency room search

Scenarios

Positive response Tom has already downloaded the mobile application called *SmartCityAdvisor* and wants to find an available emergency room in a hospital which is closest to his actual place, located in "Piazzale Gorini 18, Milano". He enters in the emergency room search section of its application, select the need/specialization, click on search and waits for a response from the service. After a few seconds the system responds indicating that the emergency room with identification code EM23 is near you and have 5 spot available in it's emergency room and shows tom the hospital address.

Negative response Tom has already downloaded the mobile application called *SmartCityAdvisor* and wants to find an available emergency room in a hospital which is closest to his actual place, located in "Piazzale Gorini 18, Milano". He enters in the emergency room search section of its application, select the need/specialization, click on search and waits for a response from the service. Immediately the system responds indicating that there are no emergency room available near you but it shows the closest emergency room with less waiting time.

Use case

Use case	**Emergency room search**
Actors	Citizen
Goals	A citizen must be able to search for a emergency room near his location
Enter condition	• The Citizen must already be registered to the service • The Citizen must already have opened the application (mobile or web) and logged in

Event flow	1. The Citizen goes to the section for searching for an emergency room 2. The Citizen click the search (and select the need/specialization) 3. The system checks the location provided by the Citizen and computes closest set of hospitals 4. The system, based on the set of hospitals, retrieves the emergency rooms 5. If there are no emergency rooms available: (a) Go back to Event flow 2 6. The system takes the closest hospital with available emergency room 7. The system computes an approximate waiting time for the Citizen 8. The system informs the Citizen of the hospital location and the waiting time
Exit condition	The citizen is going to the closest hospital

Table 3.2: Use case: Emergency room search by a Citizen

Sequence diagrams

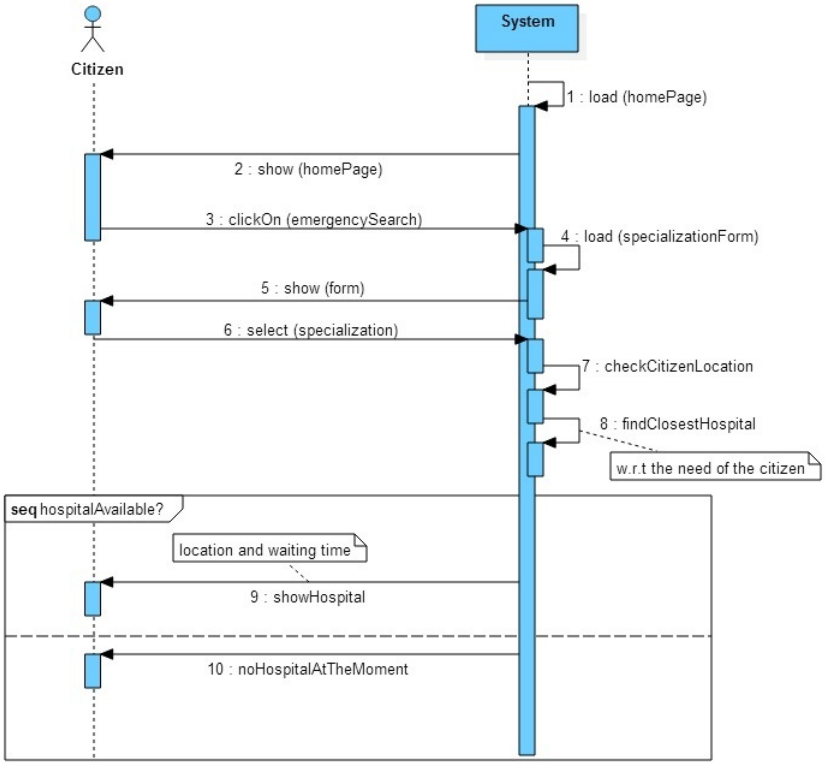


Figure 3.2: Citizen emergency room search

3.1.3 Citizen parking spot search

Scenarios

Positive response Tom has already downloaded the mobile application called *SmartCityAdvisor* and wants to find an available parking spot in city center which is closest to his actual place, located in "Piazzale Gorini 18, Milano". He enters in the parking spot search section of its application, click on search and waits for a response from the service. After a few seconds the system responds indicating that the parking spot with identification code PS221 is near you shows Tom the spot address.

Negative response Tom has already downloaded the mobile application called *SmartCityAdvisor* and wants to find an available parking spot in city center which is closest to his actual place, located in "Piazzale Gorini 18, Milano". He enters in the parking spot search section of its application, click on search and waits for a response from the service. Immediately the system responds indicating that there are no parking spot available, please try again later.

Use case

Use case	**Parking spot search**
Actors	Citizen
Goals	A citizen must be able to search for a parking spot near his location
Enter condition	• The Citizen must already be registered to the service • The Citizen must already have opened the application (mobile or web) and logged in
Event flow	1. The Citizen goes to the section for searching a parking spot 2. The Citizen click the search 3. The system checks the location provided by the Citizen and computes closest set of parking spots 4. If there are parking spots available: (a) Returns a parking spot which is near (b) Go back to Event flow 2 5. The system takes the closest parking spot available 6. The system informs the Citizen of the parking location
Exit condition	The citizen is going to the closest spot

Table 3.3: Use case: Parking spot search by a Citizen

Sequence diagrams

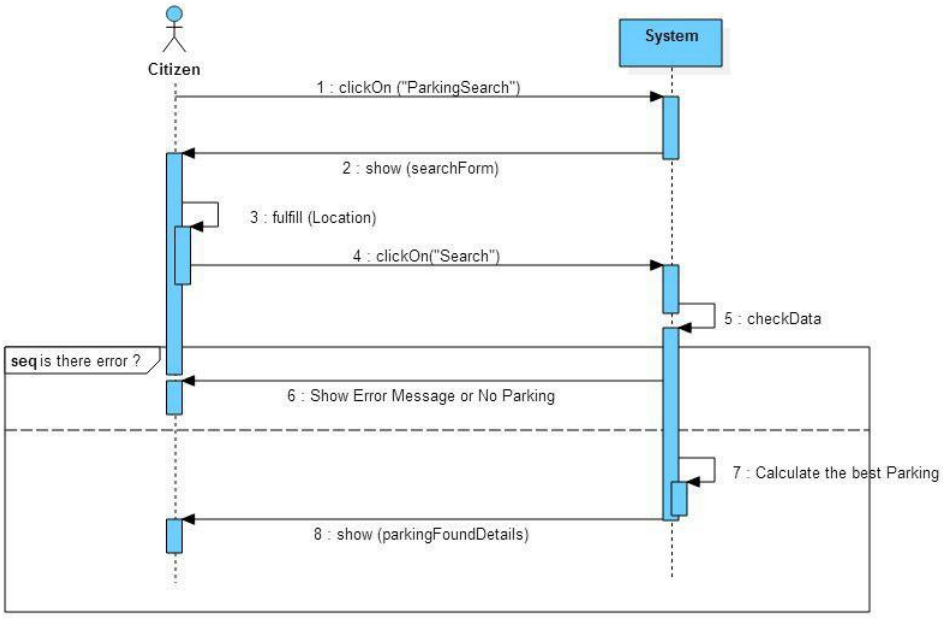


Figure 3.3: Citizen parking spot search

3.1.4 Citizen Parking Reservation

Scenario

Positive Response It is 8.00 in the morning and Tom has already downloaded the mobile application called *SmartCityAdvisor* and he has a fixed appointment at 17.00 at "Stazione Centrale of Milano" so he decides to reserve a parking spot for the 16.50 to 17:50 near the appointment address. He enters in the apposite section of the application, indicates the address, the start time and the duration. The system, after a few seconds, reply positively to Tom saying that the reservation has been accepted and the Parking identification number and it's address.

Negative Response It is 8.00 in the morning and Tom has already downloaded the mobile application called *SmartCityAdvisor* and he has a fixed appointment at 17.00 at "Stazione Centrale of Milano" so he decides to reserve a parking spot for the 16.50 to 17:50 near the appointment address. He enters in the apposite section of the application, indicates the address, the start time and the duration. The system immediately responses that the reservation cannot be accepted because there is no empty parking spot at specified address and time.

Use case

Use case	**Reserve a parking spot**
Actors	Citizen
Goals	A Citizen must be able to reserve a parking spot near specific address at a specific time and date
Enter condition	• The Citizen must already be registered to the service • The Citizen must already have opened the application (mobile or web) and logged in

Event flow	1. The Citizen goes to the section for reserving a parking spot 2. The Citizen fills the form with: location, date, time, duration 3. The Citizen submit the reservation to the system 4. If there are errors with the data (missing fields, not valid locations or duration greater than 3) or the date and time provided are not 2 hours in advance: (a) The system notifies the error to the Citizen (b) Go back to Event flow 2 5. The system checks the location provided by the Citizen and the time then retrieves the corresponding free parking spots 6. If there are no free parking spot: (a) Go back to Event flow 2 7. The system takes the first parking spot from list 8. The reservation is stored in the system 9. The system sends to the Citizen the location of parking spot and it's identification number
Exceptions	The Citizen can abort the procedure only during the waiting phase. If he decides to abort the procedure: 1. If the time of cancellation of the reservation is later than 30 minutes before the meeting time: (a) The system notifies the Citizen of the impossibility to cancel the reservation 2. The system cancel the reservation 3. The system notifies the Citizen of the successful deleting of the reservation
Exit condition	The Citizen going to parking spot and park his car successfully

Table 3.4: Use case: Parking spot reservation from a Citizen +

Sequence diagrams

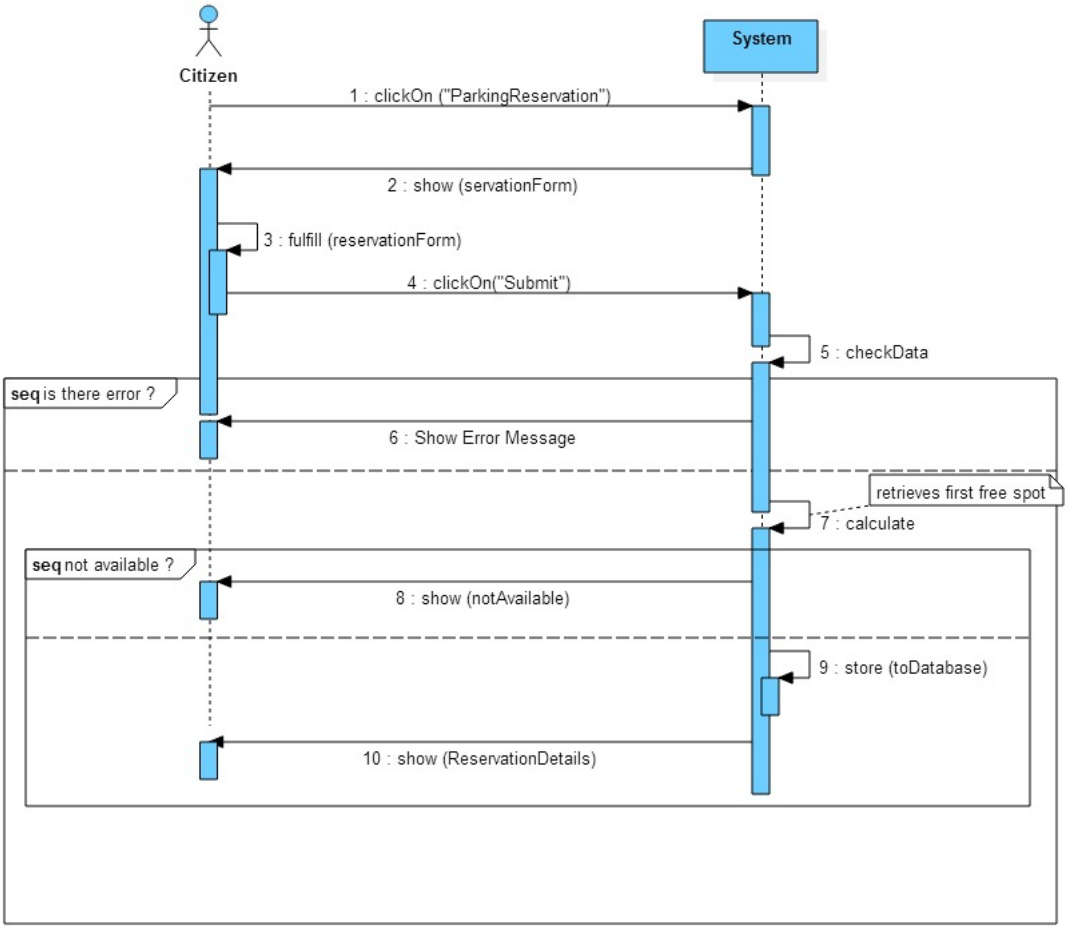


Figure 3.4: Citizen Parking Reservation

3.1.5 Control Center: limit traffic to the city

Scenario

Bob is an administrator of the control center that works in the large city and has using the application *SmartCityAdvisor - Control Center Edition*. He is in the middle of his working day and he receive a request from government to limit the traffic of city center because of visiting of a VIP group.
Bob go to the traffic limit section of the application and activate it giving the start time and duration. The system will inform citizens the new limitation start time and duration and suggesting new path that can be used through the application and the LCD s around the city.

Use case

Use case	**Limit traffic to the city**
Actors	Control Center
Goals	• Control Center must limit traffic of city center through traffic lights
Enter condition	The Control Center must have already opened the application
Event flow	1. The Control Center receives a request from government to limit traffic of city center telling the start time and duration 2. The system inform the citizens through the application and LCD s around the city about the limitation 3. The Control center setup the traffic lights to limit the traffic of city center
Exit condition	Duration of the event passed

Table 3.5: Use case: Limit traffic of city center

Sequence diagram

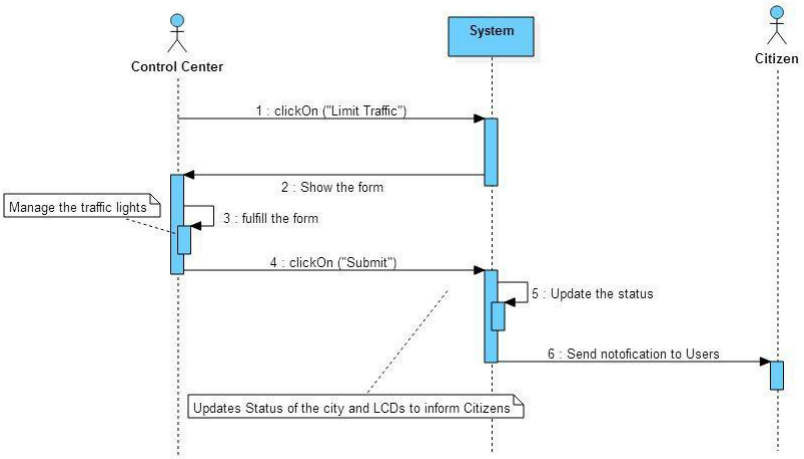


Figure 3.5: Control Center: limit traffic to the city

3.1.6 Control Center: reduce energy consumption of the city

Scenario

Bob is an administrator of the control center that works in the large city and has using the application *SmartCityAdvisor - Control Center Edition*. He is in the middle of his working day and he receive a notification from system that CO_2 emission is high in city center.
Bob go to the energy reduction section of the application and activate it giving the start time and duration. The system will inform citizens the new reduction start time and duration through the application and the LCD s around the city.

Use case

Use case	**Reduce city energy consumption**
Actors	Control Center
Goals	• Control Center must reduce the energy consumption of the city
Enter condition	The Control Center must have already opened the application
Event flow	1. The Control Center notify about high level of CO^2 and decide to reduce the city energy consumption for specific time and duration 2. The system inform the citizens through the application and LCD s around the city about the reduction 3. The Control center setup the city properties in power save mod (lights and etc)
Exit condition	Duration of the event passed

Table 3.6: Use case: reduce energy consumption of the city

Sequence diagram

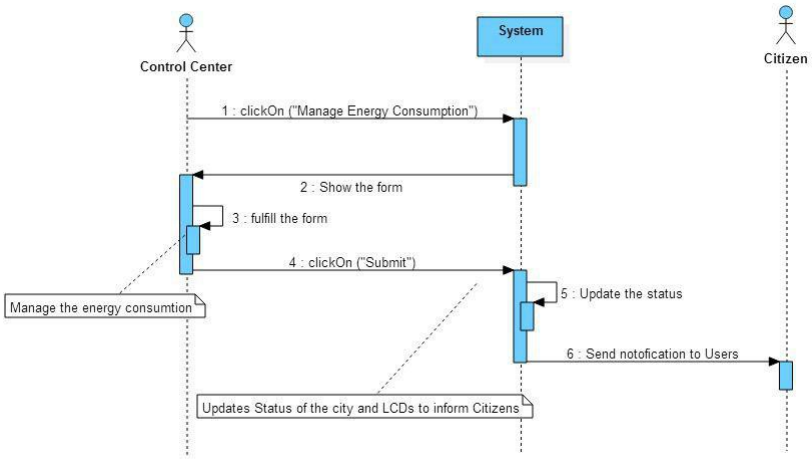


Figure 3.6: Control Center: reduce energy consumption of the city

3.2 Functional requirements

We divide the functional requirements of the application by the user class (actors) which are (mainly) involved in it:

3.2.1 Citizen

R.P.1) The system must not allow an already signed up Citizen to register himself (same username) again to the system

R.P.2) The username provided in the registration phase must not be empty

R.P.3) The password provided in the registration phase must not be empty

R.P.4) The system must notify the Citizen in case:

- The username provided is empty
- The password provided is empty
- The username provided already exists in the system

and must abort the registration procedure

R.P.5) The system must provide for the Citizen with a way to abort the registration procedure

R.P.6) A citizen request to parking spot reservation must be refused if and only if there are no parking spot available in the corresponding zone

R.P.7) A reservation must be refused if and only if:

- There are no parking spot available in the corresponding zone
- time(parking time) − time(reservation) $<$ 1 hours

3.2.2 Control Center

R.T.1) Control Center, should always available.

R.T.2) Control Center, should be notified about the new limitation 1 hour before the event

3.3 External interface requirements

3.3.1 User interfaces

Here we present the mock-ups of the application user interface.
We will present the user interface regarding only the one mobile applications (one for the Citizens and one web application for the Control center).
The web application interface for the Citizen is a natural extension of the mobile one.

The user interface must be as simple as possible: both the users must be able to use it immediately after the installation.

Home page for the Citizen

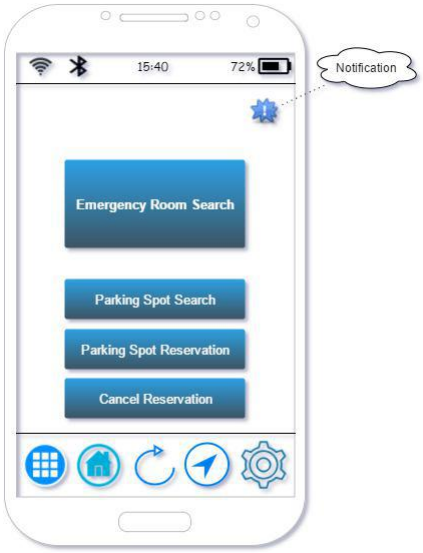


Figure 3.7: Home page for the Citizen

This screen will present to the Citizen the possibility to

- Emergency room search
- Parking spot search
- Parking spot reservation

Search for emergency room

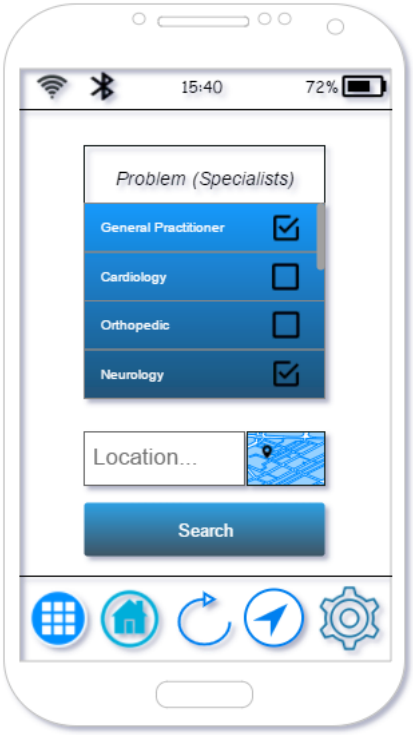


Figure 3.8: Search for emergency room

This screen is reached by pressing the button "Emergency room search" from the Home Page. The user can:

- Insert his location
- Choose his problem
- Search

Parking Search

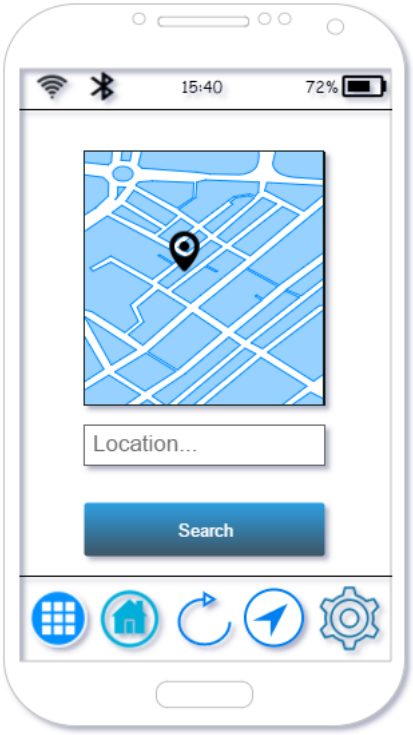


Figure 3.9: Parking search

This screen is reached by pressing the button "Parking spot search" from the Home Page. The user can:

- Insert his location
- Search

Parking reservation

Figure 3.10: Parking reservation

This screen is reached by pressing the button "Parking spot reservation" from the Home Page. The user can:

- Insert his location
- Specify the date and hour of the start parking time
- Specify the date and hour of the end parking time
- Submit the reservation request to the system

Cancel Reservations

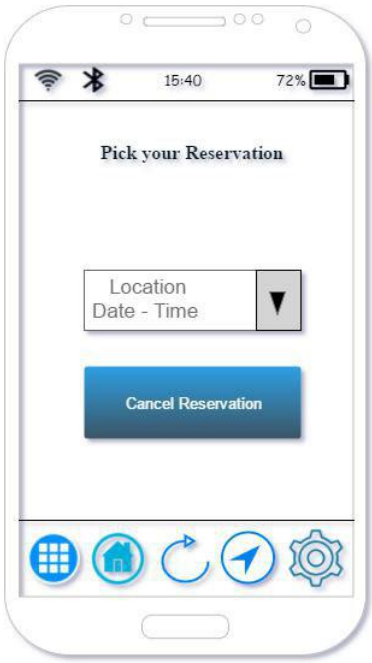


Figure 3.11: Cancel Reservations page of a citizen

This screen is reached by pressing the button "Cancel Reservation" from the Home Page. The user can:

- View all the reservations
- Cancel a reservation

The permission of canceling a reservation will be granted only if this is done at least 10 minutes before the meeting time. A pop-up will notify the citizen of the result of the operation.

Control Center Home Page

This is the first screen a Control center will see at the opening of the application:

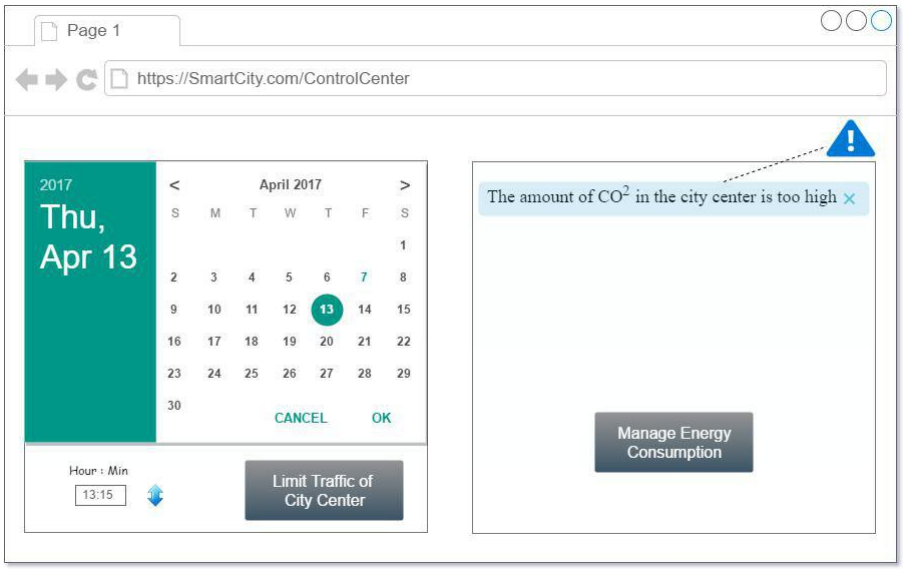


Figure 3.12: First screen of the application for Control Center

In this page the Control Center can:

- limit the traffic of city by inserting time and duration
- reduce city energy consumption

Control Center Notifications

This is the screen a Control center will see at the pop-up notifications:

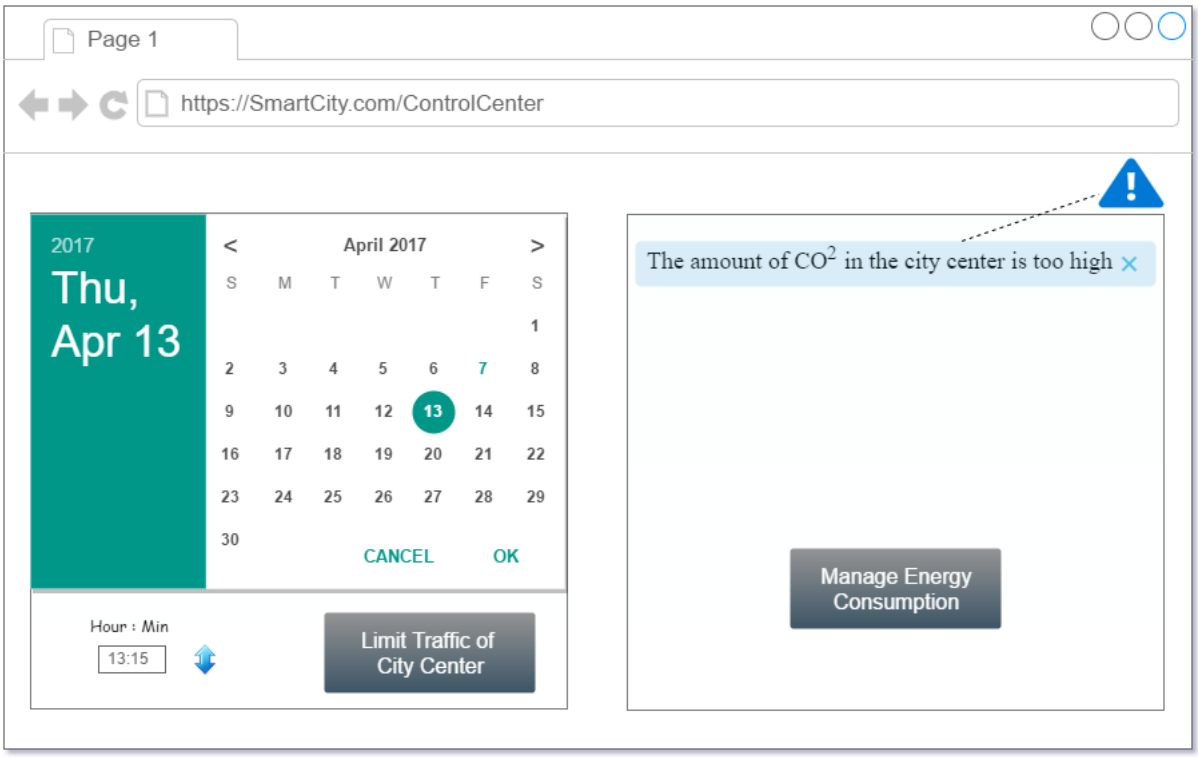


Figure 3.13: Pop-up screen of the application for Control Center

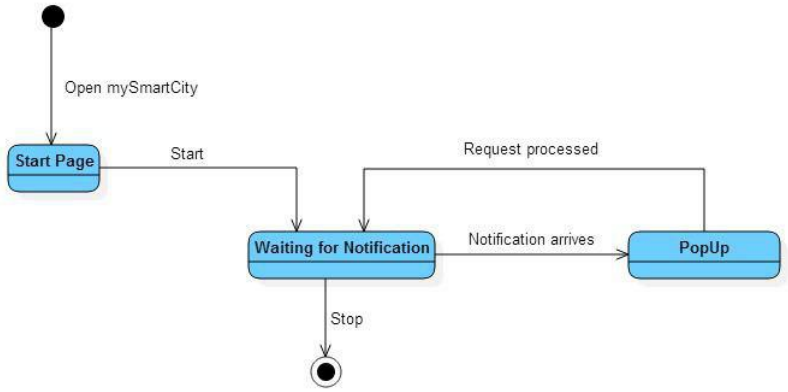
In this page the Control Center can:

- limit the traffic of city by inserting time and duration
- reduce city energy consumption

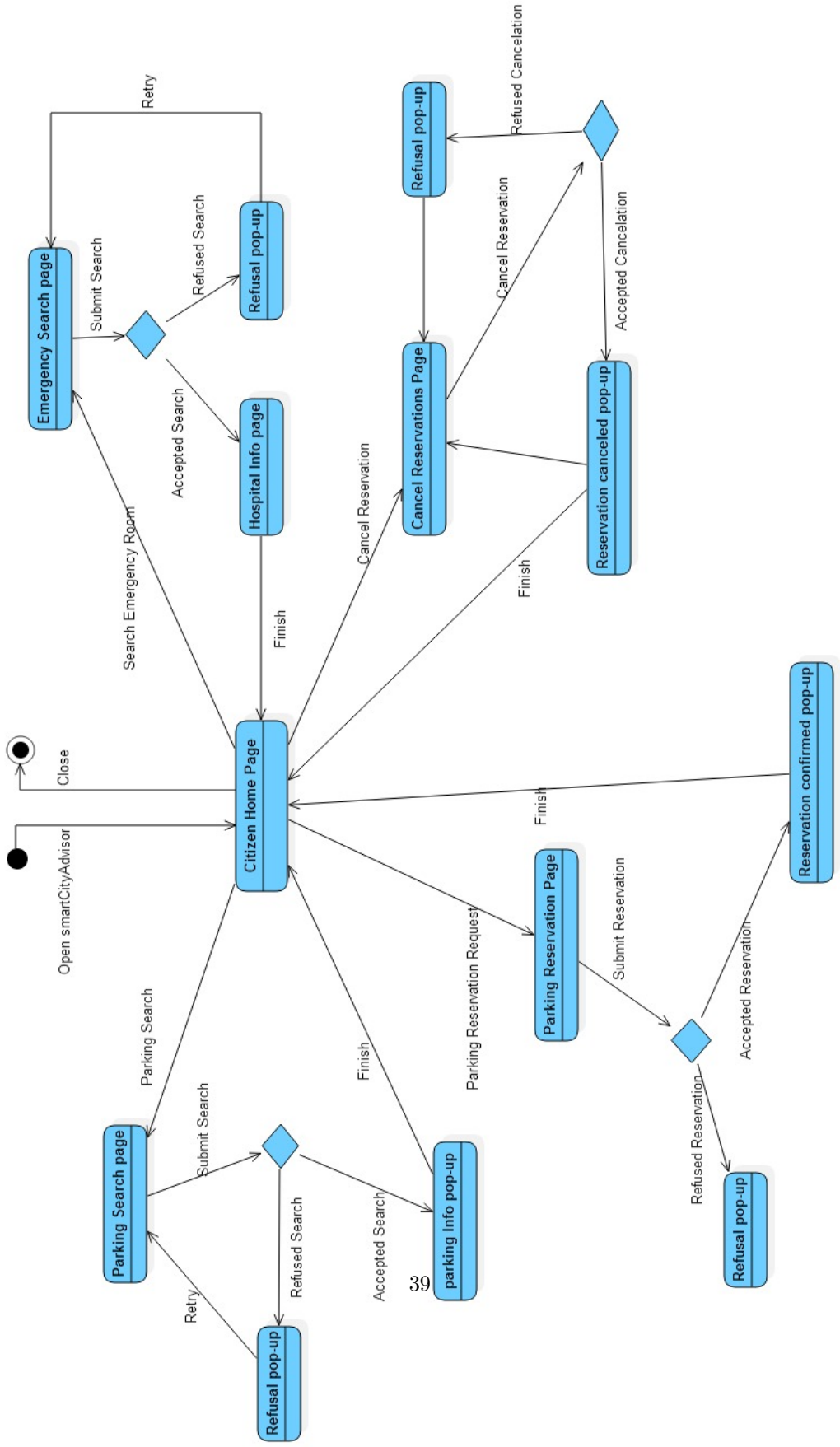
3.3.2 GUI state chart

We can summarize the behavior of the user interface through a state chart in which every state represents a specific screen of the application and each transition is basically a user input or a system communication to the application.

Control center usage of the user interface



Citizen usage of the user interface



3.3.3 Software interfaces

The software to be must provide a *programmatic interface* to enable easy future extensions.

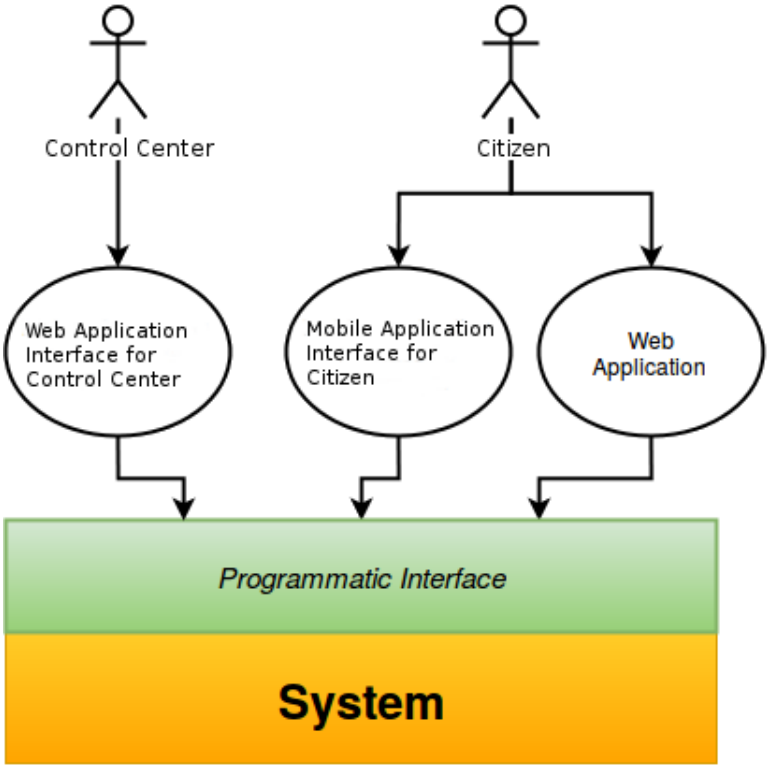


Figure 3.14: General schema of the architecture of the programmatic interface

Functional Requirements

1. The interface must have the possibility to add to the system new functionalities

3.3.4 Communication interfaces

The only communication protocol used is HTTP and HTTPS.

3.4 Non Functional Requirements

3.4.1 Performance requirements

The performance of the web and mobile applications are strictly related on the performances of the internet connection of the devices they are running on. The interfaces must not cause any delay in the interaction.

3.4.2 Availability

The system must work 24/24 7/7 with a maximum fault time of 2 hour / week only on midnights

3.4.3 Design constraints

We don't have any particular design constraint.

Chapter 4

Appendix

4.1 Alloy

For the evaluation of the model and the elicitation of the requirements we used the specification language *Alloy* which enabled us to express the structural and behavioral constraints of the software system *SmartCityAdvisor*.

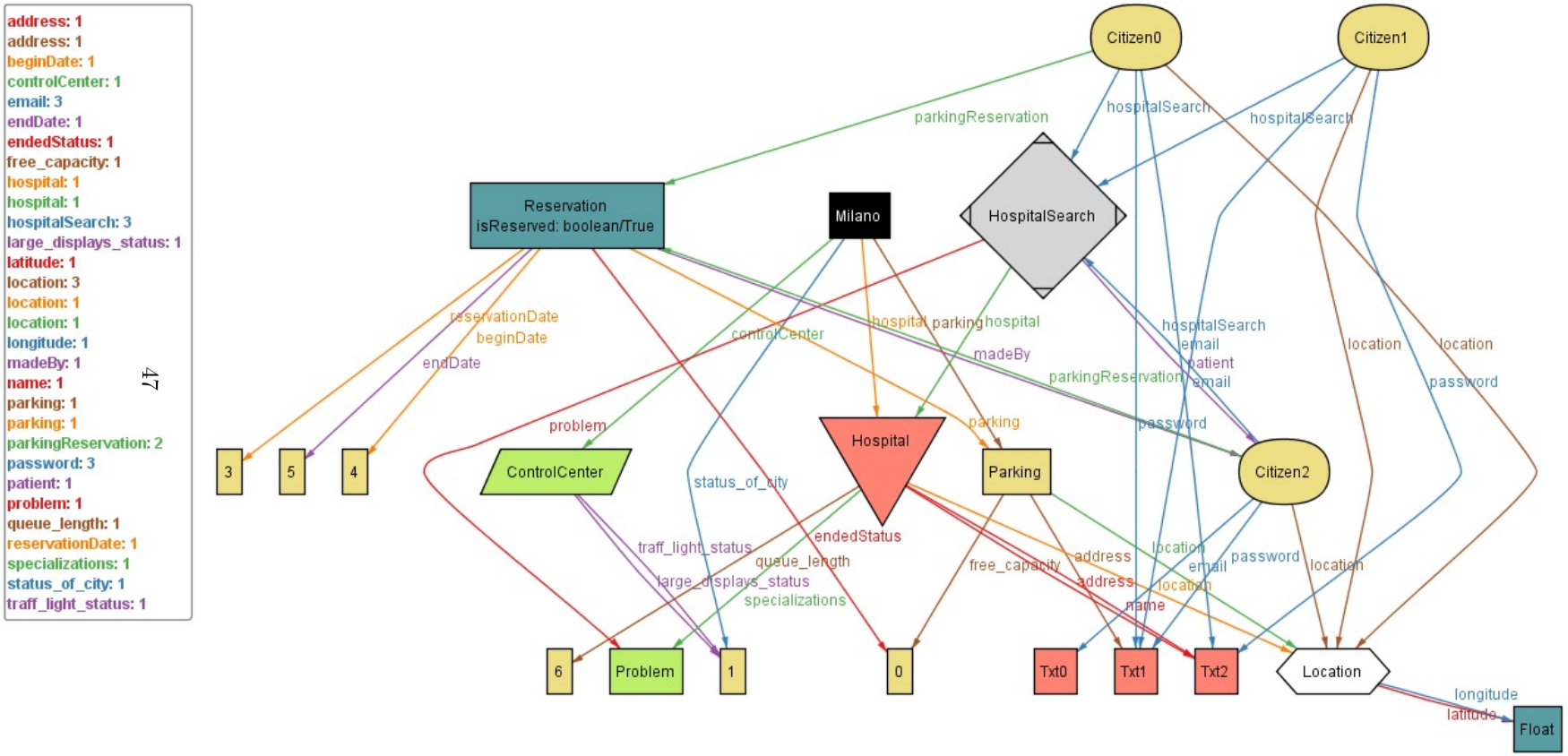
```
// Defines Bool, True, False
open util/boolean

abstract sig City{
citizen: set Citizen,
hospital: set Hospital,
parking: set Parking,
controlCenter: one ControlCenter,
status_of_city: one Int
}{
                status_of_city = 1 or
//not normal situation (VIP arrival, accident, high level of CO2, ...)
                status_of_city = 0                          //normal situation
}

one sig Milano extends City{}


sig ControlCenter{
traff_light_status: one Int,
large_displays_status: one Int,
}{
                traff_light_status=1 or
//manage traffic path to out of the city center

                traff_light_status=0            //normal path

                large_displays_status=1 or
//shows the alternative pathes and avoid to enter to the city center
```

```
                large_displays_status=0
//shows the level of CO2
}


sig Citizen{
email: one Txt,
password: one Txt,
location: lone Location,
parkingReservation: set Reservation,
hospitalSearch: set HospitalSearch
}


sig Parking{
location: one Location,
address: one Txt,
free_capacity: one Int
}
{
free_capacity >= 0
}


sig Reservation{
madeBy: one Citizen,
parking: one Parking,
reservationDate: one Int,
isReserved: one Bool,
beginDate: one Int,
endDate:one Int,
endedStatus: one Int,
}{
                endedStatus=0 or
//car has not left the parking yet or not arrived yet

                endedStatus=1                   //car has left the parking
                beginDate>0
                endDate>0
}



sig Location{
longitude: one Float,
latitude: one Float,
}
```

```
sig Hospital{
name: one Txt,
location: one Location,
address: one Txt,
queue_length: one Int,
specializations: set Problem,
}{
queue_length>= 0
}

sig HospitalSearch{
patient: one Citizen,
problem: one Problem,
hospital: set Hospital,
}



sig Txt{}
sig Float{}
sig Problem{}


//<<              FACTS            >>\\

// it is not possible to create two different profiles with the same email address
fact uniqueEmail{
all c1, c2 : Citizen | c1.email = c2.email implies c1=c2
}


// status of the City and status of traffic lights and displays
fact statusProperties{
all C: City, CC: ControlCenter |
(C.status_of_city=0 implies  (CC.traff_light_status=0
and  CC.large_displays_status=0))

and
(C.status_of_city=1 implies  (CC.traff_light_status=1
and  CC.large_displays_status=1))
}


// parking reservation properties
fact parkingReservationProperties{
all r: Reservation |
(one r.beginDate and one r.endDate) <=>  (one r.endedStatus)
// if the reservation has begin and end value
```

```
// then it is ended and the status is 1, otherwise it is not ended
// and the status is 0
}

// citizen can not make a new parking reservation
// before ending the previous one
fact noTwoParkingReservationAtTheTime{
no r1, r2 :Reservation |
r2.beginDate > r1.beginDate
and
r1.endedStatus=0
and
r1.madeBy=r2.madeBy
}


// properties for begining and ending times
fact parkingDateProperties{
all r: Reservation |
r.beginDate < r.endDate
}

fact  parkingReserveBeforeUseIt{
all r: Reservation |
r.reservationDate < r.beginDate
}

// parking must be reserved at most one hour before use it
fact  parkingTimeProperties{
all r: Reservation |
r.reservationDate + 1 >= r.beginDate
}

// consistency of patient need with specializations of the hospital
fact specailizationSearch{
all  s: HospitalSearch |
s.problem in s.hospital.specializations
// the need of the patient , must be in the list of
// the specialization of the hospital
}

assert noParkingReservationInThePast{
all r:Reservation |
(r.isReserved= True implies r.beginDate > r.reservationDate)
}
//check  noParkingReservationInThePast
//OK
```

```
pred show{
#Citizen >= 3
#Hospital = 2
#Parking = 2
#problem >= 3
}
run show
```


address: 1
address: 1
beginDate: 1
controlCenter: 1
email: 3
endDate: 1
endedStatus: 1
free_capacity: 1
hospital: 1
hospital: 1
hospitalSearch: 3
large_displays_status: 1
latitude: 1
location: 3
location: 1
location: 1
longitude: 1
madeBy: 1
name: 1
parking: 1
parking: 1
parkingReservation: 2
password: 3
patient: 1
problem: 1
queue_length: 1
reservationDate: 1
specializations: 1
status_of_city: 1
traff_light_status: 1
47
Citizen0
Citizen1
parkingReservation
hospitalSearch
hospitalSearch
Reservation
isReserved: boolean/True
Milano
HospitalSearch
reservationDate
beginDate
endDate
controlCenter
hospital
parking
hospital
madeBy
hospitalSearch
email
patient
email
parkingReservation
location
location
password
problem
parking
password
3
5
4
ControlCenter
Hospital
Parking
Citizen2
status_of_city
traff_light_status
endedStatus
queue_length
large_displays_status
specializations
free_capacity
address
address
location
name
location
email
password
location
6
Problem
1
0
Txt0
Txt1
Txt2
Location
longitude
latitude
Float

4.1.1 Notes about the model

Obviously we did not model all the requirements stated in the previous parts.
We mainly used Alloy in the first part of the elicitation of requirements in order to understand the **main constraints and relations** between the *entities* involved in our problem.
For these reasons the model above in of course incomplete and simplified. We are not interested, at this level of abstraction, in modeling every aspect of the system.

4.2 Working hours

This document was written in a total amount of around 40 hours divided so between the two group elements:

- Hamidreza Hanafi: 20 hours
- Navid Heidari: 20 hours

The two group elements were not in charge of a particular section/chapter but both worked together at the whole document.