Requirements Analysis and Specification Document

Gianpaolo Branca Luca Butera Andrea Cini

POLITECNICO
MILANO 1863

Contents

Introduction

Description of the given problem

We need to develop a system to support an electric car-sharing service, which is accessible via mobile application both on iOS and android.

Current company situation

The company which wants to provide the car-sharing service is already in the public transport business, therefore they have already a network of maintenance operators in the city area. They also have an information system which provides channels for costumer care and databases that can be used to store informations about clients and service usage. The company also has an efficient internal communication system that will be used in our system to be through the provided APIs.

Goals

We divide the goals in two sections, the first one which contains the goals achieved by modules of the application, and the second one, containing goals achieved by an embedded system installed on the vehicle.

- [G1] Allows the clients to find an available car within a selected radius around his or a specified location.
- [G2] Allows the clients to book a car and pick it up.
- [G3] Monitoring the usage of the car and charge the client with the right fare.
- [G4] Incentivizes a correct usage of the service to allow as many as possible users to use the same car without the need of the service of an operator.
- [G5] Ensures a correct distribution of cars in the recharging stations according to the available plugs.
- [G6] Allows operators to manage and monitor the state of all the cars and notifies them when maintenance is needed on a specific vehicle.
- [G7] Allows management system to set up and modify the set of areas selected as safe for parking.
- [G8] Provides a real time, interactive, pleasant and transparent user experience.

Boundaries of the system

- The system to fulfill the goals that we have identified will use the Google Maps service to locate cars,users,operators and recharging stations and to provide the clients with navigation information.

- The system will rely on PayPal as a payment system.
- The system will provide operators of the company with the information needed for the maintenance of the vehicle but won't be worried about the effective fulfillment of the maintenance task.
- The system will be able to check if a car is parked in a safe area but won't be able to check if the car is correctly parked according to laws, anyway data concerning car usage are collected and therefore it is possible to get to the physical person who committed the illicit.
- The system will detect and notify the operators when an illegal usage of the system occurs, but will not alert the police force itself.

Domain properties and assumptions

- [D1] The users are allowed to park the car they are using in every safe area and at the company's recharging stations.
- [D2] The GPS service is always available and provides always the right position.
- [D3] The system cannot prevent theft.
- [D4] Operators are properly trained by the company to use the system and correctly mark cars under maintenance as unavailable.
- [D5] The plugs availability is correctly communicated to the system by the recharging station.
- [D6] User's mobile phones are equipped with a GPS system and a camera and they are always working properly.
- [D7] The measure of the percentage of battery charge left and the estimation of the Km/% of charge ratio are correct.
- [D8] The internet connection of the cars is always working.
- [D9] The user has accepted the terms of use of the application.
- [D10] Every car is equipped with a display.

Glossary

1. Valid credential: Name, surname, birth date, driving license, PayPal account, valid e-mail address.
2. Current car details: Remaining battery, License plate number, an estimation of the remaining autonomy expressed in kilometers (calculated at average speed of 50 km/h in city traffic), the name and an picture of the car model.
3. Money saving option: An option that if on will provide the user with the information to find a suitable recharging station according to his the destination, the availability of plugs and uniform distribution of cars among the stations.
4. Safe area: area flagged by the management system as suitable for leaving the car and ending the ride.

5. Operator: in this document we refer as operator to the employees in charge of monitoring the state of the car in from dedicated terminals of the company.
6. On screen notification: is a notification which is displayed on the screen located inside the vehicle.
7. Plugged: a car is considered plugged when a sensor in the recharging station detects that the specific car has been connected to the recharging system.
8. Busy: a car is marked as busy when left parked by a client but kept booked.

Text assumptions

1. Discounts and penalties will be applied only in the case of ride not shorter than 2km, so that the system will not punish users for not using poorly charged cars for short rides and will not encourage users to use fully charged cars less to get the discount.
2. Discounts and penalties percentage values can be customized by the management system.

Actors identifying

We have two main actors:

Client: is a person who has dowloaded our application and is registered to the service.

Operator: is an employee who has access to an interface that allows him to monitor the state of cars and station and eventually send assistance.

There are also secondary actors (such as third party service providers).

Requirements

Functional Requirements

In the following section we are going to identify the requirements that our system will have to fulfill to meet reach the goals.

- [G0] Users must be able to access to the system
  - [R0.1] User must sign up with valid credential
  - [R0.2] System must generates a password for the user
  - [R0.3] User must be able to visualize and modify all his personal informations

- [G1] Allows the clients to find an available car within a selected radius around his or a specified location.
  - [R1.1] The system must retrieve the location of the user
  - [R1.2] The user must be able to specify the radius (in km) around the selected location for the car research
  - [R1.3] The system must return to the user a map with the location of all the available cars around the chosen position
  - [R1.4] Upon the selection of a car the system must retrieve an informative screen with current car details.
- [G2] Allows the clients to book a car and pick it up.
  - [R2.1] A client must be able to choose one of the available cars and, if PayPal ensures that he can pay, book it.
  - [R2.2] Once a car has been booked no others reservation can be performed by the same client until the first one is pending.
  - [R2.3] After the reservation has been confirmed to the client, he has a maximum of 1 hour to reach the car, unlock it and start the engine. If the timeout expires the reservation is cancelled and the fee is applied.
  - [R2.4] The client is able to unlock a booked car trough the app at any time after the reservation, however he has a maximum of 15 minutes to turn it on after the unlocking. If this timeout expires, the reservation is cancelled the fee is applied.
- [G3] Monitoring the usage of the car and charge the client with the right fare.
  - [R3.1] As soon as the engine starts the system must start charging the user with a fixed amount for minute and show the current price of the ride in the display of the car.
  - [R3.2] When a car is parked in a safe area and the engine is turned off the system will ask the user through the display of the car if he wants to keep the car busy for at maximum 2h, if the user select 'NO' or does nothing and leaves the car the ride is considered as ended. If the user selects 'YES' the car is marked as busy.
  - [R3.3] A user can leave the car but keep it busy with a time limit of 2 hours, during this time since the battery is not being used the management may configure a different fare. When the timeout expires if the car hasn't been picked up yet the client will be charged with the price of the ride up to that point.
  - [R3.4] A car parked in a place not marked as safe will be considered as busy, but if the client breaks the 2 hours timeout he will get fine for improper use of the service (plus the regular price for the ride). The situation will be notified to the operators that will be able to decide if the car needs to be picked or not.
  - [R3.5] If the user drives outside the boundaries of the area of the service, the system must detect it, notify it to the user at first and

apply an additional time fare as a penalty. After 30 minutes an operator will be notified of the situation.
  - [R3.6] If the signal of a car is lost for more than 10 minutes, an operator will be notified with the last known position.
  - [R3.7] After the end of the ride the user is charged with the right amount.

- [G4] Incentivizes a correct usage of the service to allow as many as possible users to use the same car without the need of the service of an operator.(Note that discounts and penalties will not be applied to short rides, further details in Text Assumption n.4)
  - [R4.1] The system will show in the display of the car a QR code that must be scanned by the user, using the application, to check in. If 2 or more users check in, in addition to the driver, a discount will be applied to the ride.
  - [R4.2] The system will apply a discount in the case a car is left with more the 50% of the battery capacity available.
  - [R4.3] The system will detect when a car is left plugged in a recharging station at the end of a ride (using the GPS sensor and the informations sent to the system by the station) and will apply a discount . If the car is left in the recharging station but not plugged within 5 minutes the discount will not be applied.
  - [R4.4] The system will detect when a car is about to be left more than 3km away from the nearest recharging station and with 20% or less battery available, will warn the client and if the client proceeds to leave the car will apply a penalty to the price of the ride.
  - R[4.5] The client will be able to select a money saving option so that the system will provide him trough the GPS navigator of the car informations to reach the available recharging station which is more suitable according to the client destination and the need of the system to distribute car uniformly among the recharging stations.

- [G5] Ensures a correct distribution of cars in the recharging stations according to the available plugs.
  - [R5.1] The system will help operators (in the case there's no need for an on place recharge) and users with the money saving option on to choose the station in which cars should be charged and left so that cars are reasonably distributed among the different stations in the city.
  - [R5.2] The amount of plugs available should be monitored and the presence of non working ones detected.

- [G6] Allows operators to manage and monitor the state of all the cars and notifies them when maintenance is needed on a specific vehicle.
  - [R6.1] The system will provide operators of the company with an interface to check the state of the cars.

  - [R6.2] Push notifications will notify when a car is need for assistance.
  - [R6.3] Cars with low battery level which are not likely to be used anymore will be flagged.
  - [R6.4] The system must provide APIs to be used to the old system that will manage the assistance action.

- [G7] Allows management system to set up and modify parameters of the system.
  - [R7.1] The system will provide an interface to select areas to mark as safe areas for parking. The selection of the locations will be possible specifying the boundaries of the areas using a map or a radius around an address.
  - [R7.2] The system will provide an interface to select the price for minute of the rides and during the busy state.
  - [R7.3] The system will provide and interface to customize the percentage of discount and penalty for the cases highlighted in the G.4 scope.

- [G8] Provides a real time, interactive, pleasant and transparent user experience.
  - [R8.1] At the end of each ride the system must notify the user with all the informations concerning the last usage, among which the total amount charged and details about eventual discounts or penalties.
  - [R8.2] If at the beginning of a ride the client is suitable for the discount of which at [R4.1], the system notifies the correct detection with an on screen notification.
  - [R8.3] At the end of a ride, if the user results parked inside a charging station, the system reminds him to insert the plug in the specific socket to get the discount of which at [R4.4] using an on screen notification.
  - [R8.4] The system eventually notifies the user with every update regarding the service, including changes in the terms and conditions document which will always have to be accepted again.

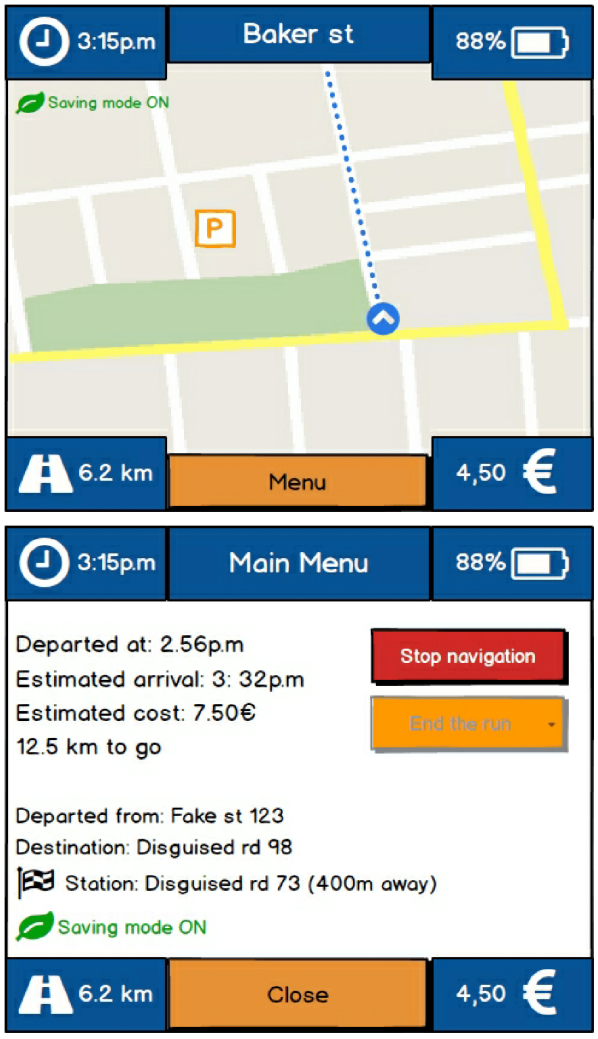
Non-functional Requirements

- The mobile application must work on all the android with version 4.3 or newer and iOS 7 or newer.
- The system must optimize bandwidth usage to guarantee a responsive service and to know the position of a car real time.
- For communication, secure protocols must be used.

Mockup

Mobile App

Car system


3:15p.m
Baker st
88%
Saving mode ON
P
6.2 km
Menu
4,50 €
3:15p.m
Main Menu
88%
Departed at: 2.56p.m
Estimated arrival: 3: 32p.m
Estimated cost: 7.50€
12.5 km to go
Stop navigation
End the run
Departed from: Fake st 123
Destination: Disguised rd 98
Station: Disguised rd 73 (400m away)
Saving mode ON
6.2 km
Close
4,50 €

Monitoring service

Scenario identifying

Here are listed some different scenarios of our system to be usage.

Scenario 1

Ted is running late for a job interview since the bus engine failed, so he opens the PowerEnJoy mobile application and searches for an available car around his current position within a radius of 700 meters. He finds that the nearest car is just 250 meters away, so he clicks the icon on the map corresponding to the chosen car and looks through the details screen; the car has 30% battery left but his destination is not far and from the estimation provided in the details he thinks the can make it, so he hits the "reserve" button, PayPal assures he can pay so the reservation is confirmed, he drops off the bus and starts walking towards the car guided by the GPS. Arrived to the car, Ted clicks the "Open Car" button and enters the vehicle. Once inside the system asks Ted to scan the QR code on the car screen with the QR scanner featured in the PowerEnJoy application. Once recognized, Ted is free to power up the car and start his ride. When Ted arrives to his destination he powers down the car and gets an on screen notification saying that he drove more than 2 kilometers, the car has less than the 20% of battery left and there's a charging station within 3 kilometers from his position, therefore he can take the car to the charging station and get a fine discount or can leave there the vehicle but this will cost him an increased fine. Ted has no time to take the car to the station so he decides to pay the penalty, he exits the car and the system takes care of locking it. The ride is successfully completed and Ted is charged for the right amount of money on his PayPal account.

Scenario 2

Gwen has invited some friends over for dinner, while eating they decide to go out for a beer later, so Gwen decides to book a car with the PowerEnJoy application, John and Paula, her two friends, are not registered to the service yet; since Gwen wants to save money by taking two registered users with her, she asks them to download the application on their smartphone and proceed with the registration. Once opened the application they're asked if they want to login or to register, they choose the second and therefore insert valid credentials and receive their password via e-mail. Finished the dinner they go out, unfortunately Gwen forgets her driving license at home so they have to go back to get it and by the time they're back the reservation time expires and Gwen is charged with a fine for not having picked up the car in time. They proceed to book the same car again, since no other did it in the meantime, to get to it and step inside. Now the three scan the QR code with their smartphones and the system notifies Gwen that a discount will be applied on the cost of the trip. Once at destination

they drop off the car and the system sends a notification warning that she left the car in a non safe area and therefore she has to keep paying with a reduced fare and must get back to the car in at most two hours. They are ok with that and walk inside the pub, again they loose track of time until Gwen's phone rings as she gets notified that two hours passed and she's getting to pay for them plus another fine for improper use of the service.

Scenario 3

Tony is preparing to go to his friend Matt, he's going to ride there with Matt's bike which he borrowed a week ago so he can give it back to him, after that they have to go to the shopping center at the boundaries of the city, so Tony decides to book a PowerEnJoy car with the mobile application. He inserts Matt's house address into the system, s and books the most convenient car. Once at Matt's place the two go to pick the car up, once they are inside and the check in is done, they decide to select the "money saving option", after inserting their destination the system provides them with a suitable charging station to leave the car at. Luckily for them the charging station is inside the shopping center area so, once parked, they power the car down and get out, once outside they insert the plug in the specific socket and leave. The system notifies Tony with the payment details including the applied discount.

Scenario 4

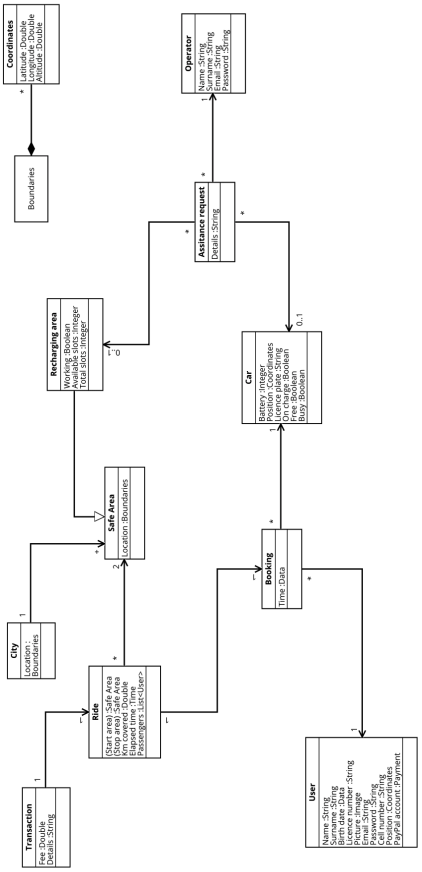
Melanie is working at the terminal, monitoring the cars in her assigned area through the interface provided by the PowerEnJoy system. She is checking through flagged cars which have low battery level. The first she overviews has been recently used and is in a central part of the city so she decides to remove the flag since it's likely it will be used anyway. The second one his in a remote part of the city and hasn't been picked up or booked since the previous day, so she decides to send maintenance personal to recharge it. To do so she enters the "assistance request" screen and start to type in the request details. While doing so she gets a push notification that warns her about the urgent need of maintenance for a vehicle. Thanks to the maintenance system integration she's able to quickly forward the request to the specific facility which will take charge of it and therefore she can go back to the request she was previously working on. After successfully sending on the request she goes back to the analysis of the flagged cars, until she gets another notification, this time the notification reports that a user improperly left a car for more than two hours outside a safe parking area, she looks over the details and finds out that the car is in the suburb of the town and so she decides to ask for an operator to have it picked and moved inside of a safe area, again through the integrated system she's able to have a maintenance operator to handle the request.

Scenario 5

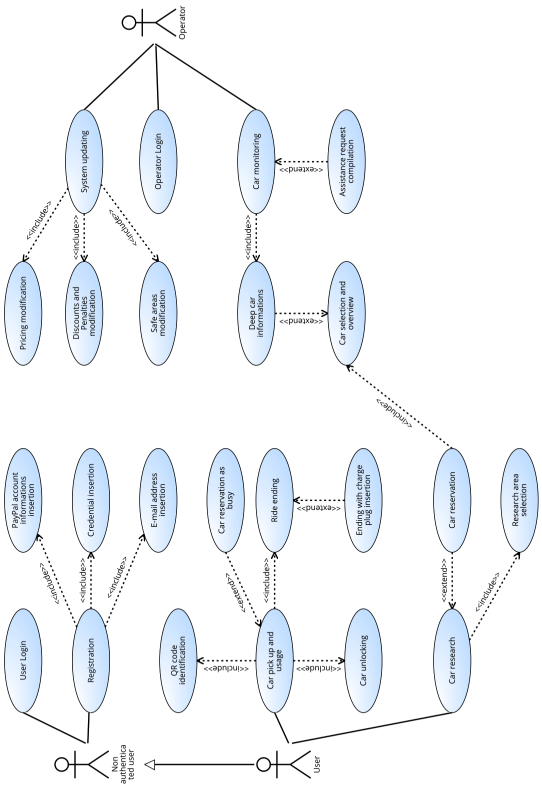
Bob, an operator of PowerEnJoy, was assigned to update the system terms regarding to a new company policy. First of all he modifies the cost per minute for using the service from 26 cents/minute to 28 cents/minute. The price raise is balanced with a proper increase in discounts, so, still through the provided interface, Bob can increase the discount, for leaving the car plugged in a charging station, from 25% to 30% on the full price of the ride. In the end Bob has to insert two new safe areas and to remove one; the interface allows Bob to select the proper utility and he can easily select, from the list of safe areas, the one to delete. Now he inserts the first new area just by specifying the chosen address and the radius around it, which in this case is of 2.6 km. Since the second area has a more complicated shape he selects the drawing tool and easily draws the polygon defining the selected area, then the system commutates the area drew on the map into proper coordinates to identify it. After this process Bob checks out the update, the system generates a proper notification for the new terms and conditions containing also the details of the update; this will be eventually endorsed by the company's legal office, and therefore sent over to the users along with the mentioned update.

Diagrams

Class Diagram



Use case diagram



Use case description

In this section are listed some common or significant use cases derivable from the Use Case diagram.

User logs in

Name: User logs in
Actors: Non authenticated user
Entry conditions: The user must be registered but hasn't logged on yet.
Flow of events:

- The user enters the login screen of the mobile application.
- The user types in his username and his password.
- The user touches the "Login" button.
- The user is redirected to the car research screen.

Exit conditions: The user is redirected on the car research screen. **Exceptions:** The username and the password are not a valid couple. If this happens the system doesn't allow the user to enter the research screen, however he's notified of the incorrectness of the credentials and therefore is kept on the login screen to try again.

User registers

Name: User registers
Actors: Non authenticated user
Entry conditions: The user is not registered to the service yet.
Flow of events:

- The user enters the login screen of the mobile application.
- The user touches the "Register" button.
- The user is redirected to the credential insertion screen.
- The user inserts his name, surname, birth date and driving license ID in any order.
- The user touches the "Next" button.
- The user is redirected to the PayPal account insertion screen.
- The user inputs his PayPal credentials.
- The user touches the "Next" button.
- The user is redirected to the e-mail insertion screen.
- The user types in an e-mail address.
- The user touches the "Confirm" button.
- The system sends a confirmation message to the specified e-mail address.
- The user is redirected to a screen informing him to check is e-mail.
- The user clicks on the link contained in the sent e-mail.
- The system activates the new user's account.

- The user is redirected into the mobile application inside a successful registration screen.
- The user touches the “Confirm” button.
- The user is redirected to the car research screen.

Exit conditions: The user is registered and is redirected to the car research screen.
Exceptions: The user inserts invalid PayPal credentials or license ID, personal information do not match license credentials, license or e-mail address are already bounded to an existing profile. In this case the user is notified of the error and is redirected to the login screen.

User searches and reserves a car

Name: User searches and reserves a car
Actors: User
Entry conditions: The user is logged into the mobile application.
Flow of events:

- The user enters the car research screen.
- The user types in an address or chooses to use his GPS location.
- The user hits the “Search” button.
- The user is redirected to the map screen showing all the available cars inside the selected area.
- The user clicks on the chosen car.
- The system show a pop-up with the car informations.
- The user hits the “Book now” button.
- The user is redirected to the map screen showing the real time route to the car and the time left before the reservation expires.

Exit conditions: The car is correctly reserved and the user is redirected to the map screen.
Exceptions: The chosen car is no more available. In this case the user is notified of the error and redirected to the map screen.

User does a full ride with money saving option

Name: User does a full ride with money saving option
Actors: User
Entry conditions: The user made a reservation for the car he’s about to ride.
Flow of events:

- The user enters his profile screen.
- The user hits the car reservation inside the history tab.
- The user is redirected to the reservation details screen.
- The user touches the “Unlock” button.
- The system unlocks the car.

- The user enters the car.
- The Car screen power on.
- The user clicks on the QR scanner button.
- The user scans the QR code on the car screen.
- The user powers the engine by pressing the physical button inside the car.
- The user clicks the "Money saving" button on the car screen.
- The car monitor show the destination insertion screen.
- The user inserts his destination and hits the "Confirm" button.
- The system calculates the optimal charge station.
- The car monitor show the GPS navigation map screen with the selected charge station as destination.
- The user rides to his destination.
- The user parks the car at the charge station.
- The user powers down the engine.
- The car monitor show the ride ending screen.
- The user selects the "End ride" button.
- The user is redirected to the plug insertion reminder screen.
- The user exits the car and closes the car door.
- The user inserts the charging plug into the car socket.
- The user is charged for the service usage.

Exit conditions: The car is parked and is being recharged. The user is charged for the payment.
Exceptions:

- In the case there's no compatible charging station the user is redirected to a screen notifying the problem and then back to the initial car screen.
- If the user parks inside the charging station but forgets to insert the plug he's notified after 5 minutes. Then he has 10 additional minutes to insert the plug, if he doesn't then the discount is not applied and he's charged for the full ride.
- The user unlocks the car but doesn't power it up in 15 minutes. If this happens then the car is marked as free and locked up again. **To decide if the user has to pay anything**

The user parks and keeps the car as busy

Name: The user parks and keeps the car as busy
Actors: User
Entry conditions: The user picked the car up.
Flow of events:

- The user parks the car and powers down the engine.
- The user is redirected to the ride ending screen.
- The user selects the "Keep as busy" button.
- The user exits the car and closes the car door.
- The system locks the car.

- The user enters his profile screen.
- The user hits the busy car reservation inside the history tab.
- The user is redirected to the reservation details screen.
- The user hits the "Unlock" button.

Exit conditions: The user picked up the car again.
Exceptions: The user doesn't unlock the car before two hours passed from the moment he made the car busy. In this case the car is marked as free and the user is charged for the extra time the car was kept busy plus a fine if the car was left outside a safe area.

Operator enrolls an assistance request

Name: Operator enrolls an assistance request
Actors: Operator
Entry conditions: The operator is logged into the system.
Flow of events:

- The operator enters the map screen.
- The operator clicks on a flagged car.
- The operator is redirected to a deep car details screen.
- The operator clicks on the "Assistance request" button.
- The system shows a pop-up screen to fill in the request.
- The operator types in the details of the request.
- The operator selects the facility of which the request must be sent to.
- The operator hits the "Send request" button.
- The system attaches the details of the car to the request and sends it.
- The operator is redirected to the map screen.

Exit conditions: The request is correctly enrolled to the specified facility. The operator is back on the map screen.
Exceptions: The request can't be enrolled or the operator forgot to fill in one of the fields. In this case the system shows an error screen notifying the problem and asking the operator to try again, then redirects him back to the request screen.

Operator inserts a new safe area

Name: Operator inserts a new safe area
Actors: Operator
Entry conditions: The operator is logged into the system.
Flow of events:

- The operator enters the safe areas management screen.
- The operator clicks on the "New safe area" button.
- The operator is redirected to a map screen showing the set of safe areas.
- The operator clicks on the polygonal drawing tool.

- The operator is redirected on the vertices insertion screen.
- The operator inserts the number of vertices.
- The operator hits the “Next” button.
- The operator inserts the coordinates of the vertices then hits the “Next” button for each one of the vertices.
- The operator hits the “Confirm” button.
- The operator is redirected to the map screen now showing the new safe area.
- The operator clicks on the “Save and exit” button.
- The system generates the update details notification.
- The system sends the update details notification to the legal office.
- The operator is redirected to the initial terminal screen.

Exit conditions: The new safe area is correctly inserted in the system and the update details notification is successfully sent to the legal office. The operator is redirected to the initial screen.
Exceptions:
- The operator inserted invalid coordinates for a vertex or an invalid number of vertices. In this case the operator is redirected to the previous screen and notified that he did something wrong and must perform the action again correctly. - The system, after ten attempts, hasn't been able to send to the legal office the update notification. In this case a warning notification containing the update information is sent to the operator and the operation is taken into account manually. **Is it ok?**

ALSO
Do we want to consider the case the new area overlaps an existing one as an exception or will the system just melt the two without generating any problem?

```
//CLASSES

abstract sig Bool{}
one sig True extends Bool{}
one sig False extends Bool{}
abstract sig CarState{}
one sig Available extends CarState{}
one sig Busy extends CarState{}
one sig UnderMaintance extends CarState{}
abstract sig BLevel{}
one sig LowL extends BLevel{}
one sig OkL extends BLevel{}
abstract sig Position{}{}
sig SafeP extends Position{}
sig UnsafeP extends Position{}
sig OutsideCity extends UnsafeP{}
sig RechArea extends SafeP{}
abstract sig AType{}
one sig Move extends AType{}
one sig Recharge extends AType{}
one sig Repair extends AType{}


sig Car{
        battery: one BLevel,
```

```
        flagged: one Bool,
        mechProbSensor: one Bool,
        status: one CarState,
        plugged: one Bool,
        position: one Position
}{
        position in UnsafeP and status≠Busy implies flagged=True
        position in OutsideCity implies flagged=True
}

sig User{
        riding: lone Car
}{riding≠none implies riding.status=Busy}

abstract sig Operator{}

sig AssistanceRequest{
        supervisor: one Operator,
        type: one AType,
        car: one Car
}

//timecode is used to simplify the model with the following meaning:
//if 2 rides has different timecodes means that they have took place
//in two disjoint period of time
sig Ride{
        driver: one User,
        passengers: set User,
        vehicle: one Car,
        timecode: one Int,
        price: one Int
        //aggiugnere posizione
}{
        price>0
        #passengers≤3
}

sig Booking{
        client: one User,
        car: one Car
}


//FACTS

fact assistanceRequestProperties{
        //A car under assistance must have be in the UndermaintanceState
        all a: AssistanceRequest | let c=a.car |( c.status=UnderMaintance
        //fatcs that determinate assistance type
        ∧ (c.mechProbSensor=True <=> a.type=Repair)
        ∧ (a.type=Recharge⟹(c.battery=LowL ∧ c.position in SafeP))
        ∧ (a.type=Move⟹ c.position not in SafeP))
        //only an assistance request for a car at a time
        all disj a1,a2: AssistanceRequest | a1.car≠a2.car
}

fact bookingProperties{
        //unique bookings
        all disj b1,b2: Booking | b1.client≠b2.client ∧ b1.car≠b2.car
        //cars booked are busy
        all b: Booking | (b.car.status=Busy
        //cars booked are not moving and users are not riding another car
        ∧ (b.client not in Car[riding])
        ∧ (b.car not in User.riding))
}

fact rideProperties{
        //users can be only in one ride at a time
```

```
        all disj r1,r2: Ride | r1.timecode=r2.timecode implies (r1.driver+r1.
            ↪ passengers+r1.vehicle)&(r2.driver+r2.passengers+r2.vehicle)=
            ↪ none
        // driver not in passengers
        all r: Ride | r.driver not in r.passengers
}

fact carsProperties{
        all c : Car |
        //cars not in use are not busy
        (notInUse[c] implies c.status≠Busy)
        //cars with a mechanical problems are put under maintence if they not
            ↪  busy
        ∧(c.mechProbSensor=True ⟹ ( c.status≠Busy ⟹ c.status=UnderMaintance
            ↪ ))
        //cars have unique positions
        ∧(all c1,c2: Car | c1≠c2 implies c1.position≠c2.position)
}

pred notInUse[c: Car]{
        c not in (User.riding + Booking.car +AssistanceRequest.car)
}

fact onlyOneDriver{
        all u1,u2: User | u1≠u2 implies u1.riding≠u2.riding
}

fact flagPolicy{
        all c: Car| (c.mechProbSensor=True or c.battery=LowL or c.position
            ↪ not in SafeP) <> c.flagged=True
}

//ASSERTIONS

assert carUnderAssistanceNotInUse{
        all c : AssistanceRequest.car | c not in User.riding
}

assert notAvailableCarWithMechProblem{
        all c: Car | c.status=Available ⟹ c.mechProbSensor=False
}

assert noTwoRide{
        all u: User, disj r1,r2: Ride | r1.driver=u∧r2.driver=u implies r1.
            ↪ timecode≠r2.timecode
}

assert onlyOneBooking{
        all b: Booking | #(Car.(b.car->b.client))=1
}

assert busyStatus{
        all c: Car | c.status=Busy ⟹(c in (Booking.car + User.riding))
}

assert carsAvailable{
        all c: Car | (notInUse[c]∧c.status≠UnderMaintance) implies c.status=
            ↪ Available
}

//fact{#User.riding>0#Booking>0 && #Ride.passengers>1 && some disj x1,x2:Ride
    ↪  | x1.timecode=x2.timecode&&#AssistanceRequest=2&& some c: Car | c.
    ↪ status=Available } // testing purpose

//PREDS
pred show{}
run show for 8
check notAvailableCarWithMechProblem
```

```
check onlyOneBooking
check noTwoRide
check carUnderAssistanceNotInUse
check busyStatus
check carsAvailable
```