

POLITECNICO DI MILANO
Department of Electronics, Information, and Bioengineering
Computer Science Engineering



Software Engineering 2

eMall - e-Mobility for All

Group components:

Roberto **CIALINI** 933385

Umberto **COLANGELO** 935073

Vittorio **LA FERLA** 224509

Academic Year 2022-2023

Contents

| | | |
|-------|---|----|
| 1 | Introduction | 1 |
| 1.1 | Purpose | 1 |
| 1.1.1 | Goals | 2 |
| 1.2 | Scope | 2 |
| 1.2.1 | World Phenomena | 3 |
| 1.2.2 | Shared Phenomena | 3 |
| 1.3 | Definitions, Abbreviations | 4 |
| 1.3.1 | Definitions | 4 |
| 1.3.2 | Abbreviations | 4 |
| 1.4 | Revision History | 4 |
| 1.5 | Reference Documents | 4 |
| 1.6 | Document Structure | 4 |
| 2 | Overall description | 6 |
| 2.1 | Product perspective | 6 |
| 2.1.1 | Scenarios | 6 |
| 2.1.2 | Class diagram | 8 |
| 2.1.3 | State Charts | 9 |
| 2.2 | Product functions | 11 |
| 2.2.1 | Let Drivers know the charging stations nearby, their cost, special offers available and book a charge | 11 |
| 2.2.2 | Let Drivers start the charging process, be notified when it ends and pay for the service | 12 |
| 2.2.3 | Let Administrator change the <i>DSO</i> , monitor the performance and statistic for every station and modify energy mix, price and special offers for every station | 13 |
| 2.3 | User characteristics | 13 |
| 2.4 | Assumptions, dependencies and constraints | 14 |
| 2.4.1 | Domain assumptions | 14 |
| 3 | Specific Requirements | 15 |

| | | |
|-------|---|----|
| 3.1 | External Interface Requirements | 15 |
| 3.1.1 | User Interfaces | 15 |
| 3.1.2 | Hardware Interfaces | 15 |
| 3.1.3 | Communication Interfaces | 15 |
| 3.2 | Functional Requirements | 17 |
| 3.2.1 | Mapping on Goals | 19 |
| 3.2.2 | Use cases | 19 |
| 3.2.3 | Use case diagrams | 27 |
| 3.2.4 | Sequence diagram | 30 |
| 3.2.5 | Mapping on requirements | 46 |
| 3.3 | Performance Requirements | 46 |
| 3.4 | Design Constraints | 46 |
| 3.4.1 | Standard compliance | 46 |
| 3.4.2 | Hardware limitations | 47 |
| 3.5 | Software System Attributes | 47 |
| 3.5.1 | Reliability | 47 |
| 3.5.2 | Availability | 47 |
| 3.5.3 | Security | 48 |
| 3.5.4 | Maintainability | 48 |
| 3.5.5 | Portability | 48 |
| 4 | Formal Analysis | 49 |
| 4.1 | Alloy Code | 49 |
| 4.1.1 | First model | 57 |
| 4.1.2 | Second model | 57 |
| 4.1.3 | Dynamic model | 58 |
| 5 | Effort spent | 59 |
| 5.0.1 | Roberto Cialini | 59 |
| 5.0.2 | Umberto Colangelo | 59 |
| 5.0.3 | Vittorio La Ferla | 59 |
| 6 | References | 60 |

1 Introduction

1.1 Purpose

Nowadays sustainability is one of the most important and debated topics in our society. In fact, in the next few years we are going to deal with a huge green transition to limit our carbon footprint on the planet, such as in the transportation field, which finds itself as one of the main contributors of global warming. In this direction, in recent years the old motor vehicles running on gasoline are leaving space for electric powered vehicles, even though there are several central aspects to deal with in order to let the electric vehicles be competitive with the old vehicle generation.

In this direction, the goal is to create a fully operative and diffused infrastructure for the fast charging of the batteries, which is one of the main limitations for the final customer. In fact, the batteries need to be charged often and nowadays the task of finding an available charging spot is not as easy as it seems.

With this issue in mind, *eMall* is an operating system, itself composed of two sub-systems, whose goal is to offer a way to find available charging stations for electric vehicles, offering at the same time to the user the possibility to access to several features such as the reservation of a specific socket at a certain timeframe or the reception of personalised proactive suggestions by the system.

This document will further explain in detail goals and requirements put on the system to be with this purpose of guiding the development.

1.1.1 Goals

| Requirements | Description |
|--------------|---|
| G1 | Allow Drivers to find available charging stations, the nearest ones, their energy cost per unit (MW/h), any special offer they have, their available sockets and a path to reach the selected one |
| G2 | Allow Drivers to book the desired charging station for a certain timeframe and possibly cancel it |
| G3 | Allow Drivers to start and monitor the charging process at a certain station and to be notified when the charging process is finished |
| G4 | Allow Administrators to track data regarding internal and external status of each charging station |
| G5 | Allow the Administrators to manually or automatically decide for each station where to get energy for charging (station battery, <i>DSO</i> , or a mix) |
| G6 | Allow Administrators to have access to <i>DSOs</i> information about the current price of energy and to decide from which to acquire energy |
| G7 | Allow Drivers to receive recommendations based on the status of the battery of his active vehicle, the schedule of the user and his current position |

1.2 Scope

While there are several stakeholders to consider, this document is only concerned about two actors: Drivers and Administrators. The Driver is the final user, the one who interacts with the *eMSP* to have the possibility of booking recharges for his vehicles. Instead the role of the system Administrator mainly concerns to monitor the correct behaviour of the system and to take strategic decisions.

The main system is divided into two subsystems: the *eMSP* and *CPMS*. The *eMSP* is designed to be an interface and to communicate both with the Driver and the Administrator, driving their requests. The *CPMS*, instead, is modelled on the *OCPI 2.2.1* protocol and is used to manage the IT infrastructure of a specific *CPO*. The main task of the *CPMS* is to supply information about its *CPO* charging stations to the *eMSPs* it is linked to, both for the Driver and the Administrator usage.

Although *CPO* and *DSO* are mentioned in this document along with the other entities described before, we will not consider either their internal system or their decision

making.

This application is supposed to work properly in every situation in which are well defined the previously mentioned rules, with no limitation to the metropolitan areas.

1.2.1 World Phenomena

| Identifier | Description |
|------------|--|
| WP1 | Charging stations are owned by a <i>CPO</i> , can have a stationary battery and provide energy through sockets of different speeds |
| WP2 | <i>DSOs</i> provide energy to <i>CPOs</i> stations |
| WP3 | Drivers use the charging stations to charge their vehicle |
| WP4 | <i>CPO</i> Administrators manage their <i>CPMS</i> |

1.2.2 Shared Phenomena

| Identifier | Description |
|-------------|--|
| SP1 | User registers an account |
| SP2 | Users pays the cost for charging |
| SP3 | User chooses the charging station he prefers from the available ones |
| SP4 | Administrators have access to the all the data regarding their <i>CPMS</i> |
| SP5 | User books for a specified amount of time a socket in a charging station |
| SP6 | System visualises data about recharging station based on the <i>CPO</i> information |
| SP7 | User decides to end the charge of the battery |
| SP8 | System makes recommendation on the best charging station available |
| SP9 | System sends notification to user |
| SP11 | Administrators manually or automatically select from which <i>DSO</i> to acquire energy from |
| SP11 | Administrators manually or automatically select for each station where to get energy |
| SP12 | System is notified when the battery is fully recharged |

1.3 Definitions, Abbreviations

1.3.1 Definitions

| Definitions | Description |
|---------------------------|---|
| Driver Identifier | To identify a specific driver, this could be an identification number such as her/his SSN |
| Car Identifier | To identify a specific car, this could be the licence plate |
| Station Identifier | To identify a specific charging station |

1.3.2 Abbreviations

| Abbreviations | Definitions |
|----------------------|--|
| RASD | Requirements Analysis and Specification Document |
| WP | World Phenomena |
| SP | Shared Phenomena |
| GX | Goal number X |
| DX | Domain assumption number X |
| RX | Requirements number X |
| GPS | Global Positioning System |
| <i>eMall</i> | e-Mobility for All |
| EV | Electric Vehicle |
| Driver | Electric vehicle driver |
| Administrator | <i>CPO</i> administrator |

1.4 Revision History

Version 1.1.0

1.5 Reference Documents

The specification document "Assignment RDD AY 2022-2023_v3.pdf"

1.6 Document Structure

This document is composed of six sections, detailed below.

In the first section the problem is introduced together with the goals of the project. Additionally, the scope of the project is specified along with the various phenomena occurring. Lastly, the necessary information to read the report is presented, such as definitions and abbreviations.

Section two contains an overall description of the system, including a detailing of its users and main functions. Moreover there is the class diagram, descriptions of several scenarios, some statecharts and finally the domain assumptions made in this report.

In section three the requirements on the system are specified. This includes functional requirements, non-functional requirements and requirements on external interfaces. Furthermore use cases are described, with accompanying use cases and sequence diagrams. Section three also contains mappings of functional requirements to the goals of the system and to the use cases.

Section four contains a formal analysis with the help of Alloy. Together with the Alloy code, the analysis objective is described.

In section five is shown the total effort spent by each of the project members.

Section six contains the references used.

2 Overall description

2.1 Product perspective

2.1.1 Scenarios

1. Electric vehicle Driver starts using the system

The electric vehicle driver Mike wants to register to the service and have access to the several facilities it offers, such as finding a charging station and charging his vehicle outdoors. He launches the service and chooses to sign up, fulfilling the mandatory information required to access the service (name, surname, email address, password and licence plate of his vehicle) and accepting the localization and calendar access conditions.

2. Electric vehicle Driver setting personalised data

Mike, an electric vehicle driver, once he has registered to the system using his credentials, by selecting his profile and then from the section “Change active vehicle”, chooses one of his vehicles and set it to be his Active Vehicle. In this way, the system automatically filters the stations according to the vehicle information, meaning that will show as clickable only the stations that have the socket compatible with his Active Vehicle.

3. Electric vehicle Driver wants to book a charging station

Mike is an electric vehicle driver who needs to charge his car. After he logs in into the system using his user credentials, he has access to the homepage where all the stations available are displayed. The user can also decide to filter the station by selecting the feature he needs the most. Once John has selected the station, he can select an available socket and timeframe. He then clicks on “Book Now” to reserve the slot for a certain timeframe. Jack, another user that needs to charge his vehicle in the same period of time, won’t be able to book the same socket and will see it as already engaged.

4. Electric vehicle Driver wants to start the charging process

Lucy, an electric vehicle driver, wants to charge her car. Once she has reached the

parking spot and has plugged in the vehicle, she logs into the system with her user credentials and by clicking on her reservation, in the section “My Reservations”, by selecting “Start Charging”, she confirms to begin the charging process. Lucy can also monitor in live the progress.

5. Electric vehicle Driver pays for the charge

Mike is logged into the system and has received the notification of the complete charge of his vehicle. Afterwards, he receives the notification of the correct payment, using the Active Payment Method previously selected in his profile.

6. Electric vehicle Driver accepts a recommendation

Mike, after logging into the system, has access to the “My Recommendations” page, where the system recommendations are shown. After selecting a recommendation he likes, he clicks on “Accept Recommendation” to book the corresponding reservation. After that, the reservation will be shown in the “My Reservations” page, along with the others he previously created.

7. CPO Administrator monitoring station status

A *CPO* employee, with station administration tasks, logs into the system using his administrator credentials. In the main page he has access to the stations of his company connected to the system back-end infrastructure. By selecting one of them, he can monitor the status, such as electricity and free slots available, current prices and if the station is properly working. He can also analyse the performance of the station in the last 6 months.

8. CPO Administrator taking decisions

The stations administrator Mike logs into the system using his administrator credentials. After he has selected a charging station, Mike can visualise its performance, can choose the energy mix used to charge the vehicles between the *DSO* and the one retrieved from the stationary battery (if present). Moreover, Mike can modify the *DSO* from which to take energy, analysing the offered price. The administrator can also choose to set these decisions to be automatized by the *CPMS*.

2.1.3 State Charts

1. Sign In and Sign Up

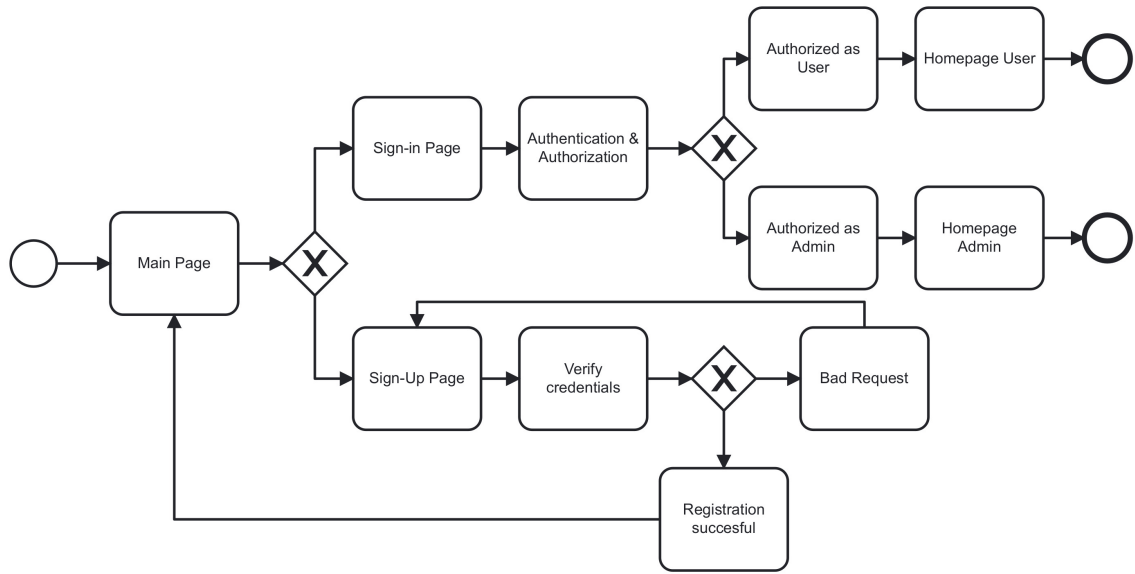


Fig. 2: Sign In and Sign Up.

The state diagram summarises the sign-in and sign-up actions. Once the Driver enters the Main Page, he can decide if he wants to register or to login, while the Administrator can only login. If the user chooses the login, depending on his authorizations, he is redirected to the Driver or Administrator main page.

2. Book a station

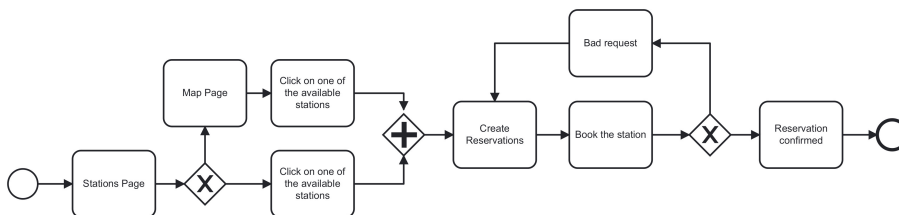


Fig. 3: Driver book a station

The EV Driver can decide which station he wants to book from two different pages. The Home Page shows a list of the available stations that can be sorted by distance, price, while the Map Page displays all the available stations on the map, including the Driver position.

3. Driver reach a station

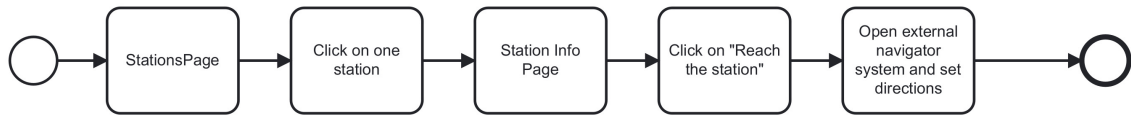


Fig. 4: *Reach a station*

Once the Driver is logged in, the first page displayed is Station Page. Afterwards, the Driver can set on his device's Maps the path to reach the station, clicking on the devote button on the station he wants to reach.

4. Administrator access to the system

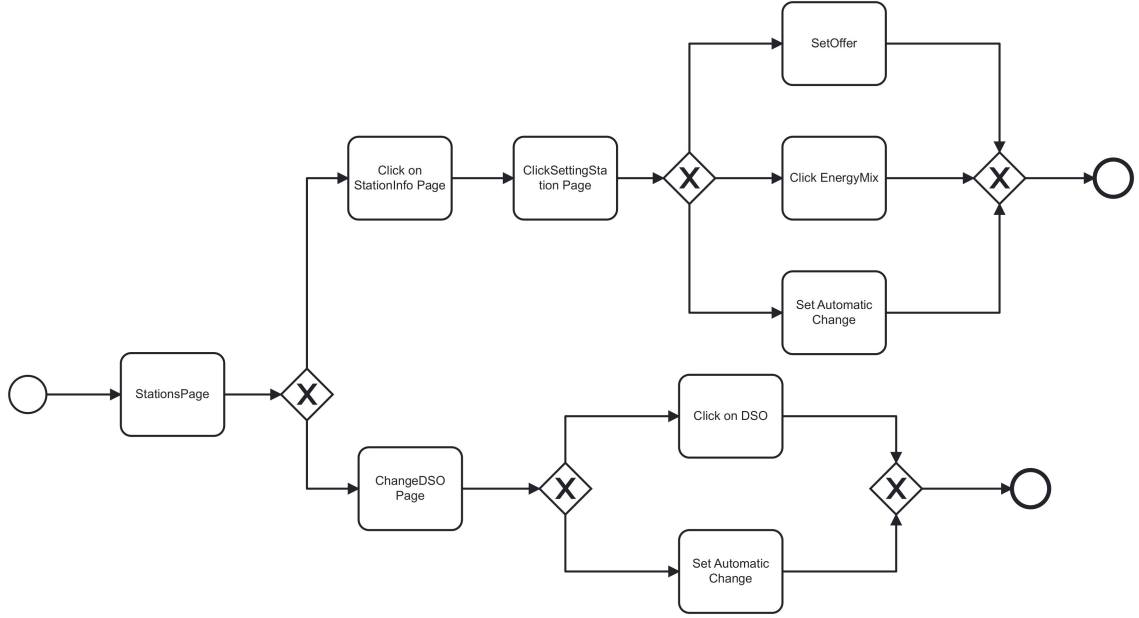


Fig. 5: Administrator access to the system

Here are described all the possible actions that the Administrators can do. The Administrator section has two main pages, the Stations Page where, clicking on a specific station there are displayed the main statistics and information (retrieved from the *CPMS* and processed by the *eMSP*). Then, he can manage the single station settings (offers, energy mix). Otherwise, he can go to the ChangeDSO Page.

2.2 Product functions

In this section the main functionalities of *eMall* are described in more detail:

2.2.1 Let Drivers know the charging stations nearby, their cost, special offers available and book a charge

One of the most important aspects of the system is the ability of the drivers to know the location of nearby charging stations: this will be achieved throughout the visualisation

of an interactive map shown in the “Map Page”, that thanks to the driver’s GPS, will show the current user location as well as the real time data regarding the charging stations nearby. In the ”Main Page” a list of all available charging stations is shown. The user can sort the charging stations by applying filters to the list, such as cost, travel distance and presence of special offers available. The user will be able to select a charging station to obtain more details regarding it and if it has an available socket, he will be able to book it for a charge in a certain time frame, depending on the ones available. After booking a charge, the user will have a certain period of time to show up at the charging station and start the charging process, otherwise the reservation previously booked will be cancelled. In case a special offer is present, the user will be notified through the presence of a special message on the charging stations promoting the offer. Furthermore, when selecting a charging station, the offer details will be shown.

2.2.2 Let Drivers start the charging process, be notified when it ends and pay for the service

After a driver reaches the station and parks his EV in the designated area, he can connect it to the charging station’s socket he previously booked. After selecting the corresponding reservation, the driver can click the ”Start Charging” button to start the charging process. After checking that the user has a valid payment method linked to his account, the charging process starts. The user will be able to monitor the current status of the battery and the estimated time left for the charging to complete. The user will be free to stop the charging process anytime through the app. After the charging process ends the payment will be automatically debited from the user’s account. When the vehicle battery is completely charged, the user will be notified through a notification on his device and will have a certain period of time to disconnect his EV and leave the charging spot free.

2.2.3 Let Administrator change the DSO, monitor the performance and statistic for every station and modify energy mix, price and special offers for every station

The system is designed to allow a simultaneous interaction between different CPO Administrators with their own *CPMS*, by which are stored and retrieved data about the stations they manage. The Administrator has the possibility to change the *DSO* from which to acquire energy or to let the *CPMS* to automatically set the energy supplier, based on the cost of electricity. Moreover, the Administrator has the possibility to manually or automatically set energy mix, price and special offers for every single station he manages. The Administrator can also visualise informations and performance of a certain station, like the presence of a stationary battery, availability and type of the sockets and other details useful in the decision making process, such as average customers usage, average energy price and average energy mix of the last 6 months.

2.3 User characteristics

The following three actors are considered in the *eMall* systems.

1. Unregistered electric vehicle driver

A driver that needs to register to the *eMall* platform before being able to use any of its functionalities.

2. Electric vehicle driver

A registered user that uses the system to find the charging stations nearby, their cost and any special offer they have. He can also book a charge for a certain timeframe and monitor the live status of the charging process

3. CPO stations administrator

A registered user, working as stations administrator for a specific *CPO*. He is able to monitor the status of his charging stations, to take decisions about the energy supplier of the stations and to manage the offers shown to the customers for each station.

2.4 Assumptions, dependencies and constraints

2.4.1 Domain assumptions

| Identifier | Description |
|------------|---|
| D1 | There exists an API where user credentials can be verified (licence plate,email..) |
| D2 | There exists an API where the correct map and driver device GPS can be retrieved |
| D3 | There exists an API where updated <i>DSO</i> prices can be retrieved and the <i>DSO</i> can be selected for energy acquirement |
| D4 | When a Driver disconnects his EV from the socket he utilised, he immediately leaves the parking area. |
| D5 | Users insert their personally identifiable information into the system |
| D6 | <i>CPO</i> Administrators have access to an already existing account on <i>eMall</i> |
| D7 | Users give the system the authorization to access personal data regarding their EV, current position and calendar |
| D8 | Energy is supplied correctly from <i>DSOs</i> to <i>CPOs</i> charging stations |
| D9 | Users give the system the authorization to access personal data regarding their EV, current position and calendar |
| D10 | The Drivers respect the reservations made by showing up at the charging station booked on time. |
| D11 | All the vehicles are supposed to have a Universal Socket accepted by all the sockets |
| D12 | Users give the system the authorization to access personal data regarding their EV, current position and calendar |
| D13 | The Drivers own the vehicles they use |
| D14 | The sockets, during the charge of a vehicle, can retrieve in real time correct data regarding the current charging speed and battery vehicle status |
| D15 | Vehicles can only use sockets of the same or lower energy power (vehicles with the rapid charge can use all types, vehicle with the fast charge can only use fast and slow sockets, vehicle with slow charge can only use slow sockets) |

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The user interface of *eMall* is both a web and a mobile application that will be used both by EV Drivers and *CPO* Administrators. It should be available as much as possible and easy to use, in particular the mobile app interface, where Drivers search and make reservations for charging slots, has to be fast and user-friendly. On the other side, Administrators are supposed to have office PCs where to work on the system, so the web application has to be optimised and oriented towards data analysis through the use of specific accurate tools.

3.1.2 Hardware Interfaces

The system sets up from the necessity to handle multiple and simultaneous commands from different actors, which are driven in their decisions by the continuous upgrade of specific information from communication sources. Moreover, since the system has to be fully available both on the mobile and on the web application, the only hardware interface requirements are a web browser (or even better a mobile application store) and the possibility to provide geolocalization information to have access to all the offered features. The system also relies on the use of different sensors deployed to obtain data, such as vehicle battery percentage or the internal status of the charging stations. This data are supposed to be managed externally and it is assumed that there are specific APIs to retrieve data for *eMall*.

3.1.3 Communication Interfaces

The system is based on the management of many information sources to provide lots of its functionalities. Therefore, it is required to communicate with external information providers from where *eMall* retrieves data or services. In particular, there are different interfaces that *eMall* exploits, possibly through Web APIs, as previously explained in the assumptions.

1. Retrieval of data on EV battery status

The interface is able to respond with the current status of the battery, if there are malfunctions or not, the battery level both during the charging process and during its usage (in line with the goal of customization and the recommendations the system gives to the user).

2. Retrieval of data on charging station internal status

The interface is able to respond with the current internal status of the charging station, which means the mix of energy used (specifying the percentage from each source), the level of the storage of the stationary battery and the availability of each slot with its main information, such as the power provided (super fast, fast, slow).

3. Retrieval geolocalization data of EV Driver

The interface is able to provide in real time the current position (latitude and longitude) of the user device in the world map.

4. Retrieval of current world map

The interface is able to provide data regarding the map through which the system shows to the user its current location, the location of charging stations and a possible path to reach them.

5. Retrieval of data on EV driver schedule

The interface provides the information about the user's daily schedule, contained in his own calendar. Using this information, the system is able to give proper suggestions, about the best time in which to charge the EV, to the user.

6. Payment authorization and correct completion

The interface deals with the payment process, in particular with the validation of the credit/debit card, the authorization to proceed with the payment (given by the user through his online bank or similar methods) and the effective correctness of the transaction.

Moreover, there are communication protocols needed for the correct forwarding of in-

formation among the different subsystems and the physical infrastructures (charging stations and its components).

1. OCPI

This protocol is required in the communication between our subsystems: *eMSP* and *CPMS*. It is mainly used to provide charging station information (socket status and details, location and tariff), to book and to authorise the charging session.

2. OCPP

This protocol is required in the communication between *CPMS* and a specific charging station. The main information forwarded are the principal operative commands of the charging session (start/stop and current status) and diagnostic information/updates.

These protocols are supposed to work properly. Furthermore, the *OCPP* specifications, differently from the *OCPI* ones, are no longer described and modelled in this document.

3.2 Functional Requirements

| Requirement | Description |
|-------------|---|
| R1 | The system shall allow an unregistered user to register an account |
| R2 | The system shall allow a registered Driver to insert one or more EV and to set one of them as active |
| R3 | The system shall allow a registered Driver to insert and modify one or more valid payment methods |
| R4 | The system shall allow a registered Driver to visualise the available charging stations in the list view (sorted by the filter he selected) or in the map view (together with his current position) |
| R5 | The system shall allow a registered Driver to view the available charging stations (sorted by the filter he selected), visualising its <i>CPO</i> owner, the current energy price, the distance from the user location and the estimated time to reach it |
| R6 | The system shall allow a registered Driver to get information (<i>CPO</i> owner, current energy price, distance from the user location and estimated time to reach it) of a station clicking on it in the map view |
| R7 | The system shall allow a registered user to book a socket at a certain timeframe |
| R8 | The system shall allow a registered user to choose the desired |

| | |
|------------|--|
| | type of charging socket (slow, fast, rapid) |
| R9 | The system shall allow a registered user to view his reservations |
| R10 | The system shall allow a registered user to get the path to reach a station starting from his reservation view or from the station view (list and map) |
| R11 | The system shall allow a registered Driver to start the charging process and to stop it |
| R12 | The system shall allow a registered Driver to view the live status of the charging process and the remaining estimated time |
| R13 | The system shall notify the registered Driver when the charging process is finished |
| R14 | The system shall allow a registered Driver to view system's recommendations |
| R15 | The system shall allow a registered Driver to accept one of the recommendations |
| R16 | The system shall allow a registered Driver to cancel a reservation |
| R17 | The system shall allow a registered Administrator to view the status of his <i>CPO</i> charging stations, including availability, <i>DSO</i> , energy price (for the <i>CPO</i>) and special offers |
| R18 | The system shall allow a registered Administrator to view status, type, availability and special offers (if present) of each socket of a certain charging station |
| R19 | The system shall allow a registered Administrator to view status, the status of the stationary battery (if present) of a certain charging station |
| R20 | The system shall allow a registered Administrator to view the current energy mix (ground, stationary battery) used in the charging process in a certain station |
| R21 | The system shall allow a registered Administrator to enable the <i>CPMS</i> to automatically select active <i>DSO</i> and energy mix, energy price (for the costumer) and special offers at a certain charging station |
| R22 | The system shall allow a registered Administrator to manually select active <i>DSO</i> and energy mix, energy price (for the costumer) and special offers at a certain charging station |
| R23 | The system shall allow a registered Administrator to view statistics for a certain charging station, including average customers usage and in which time slots, average energy price (both for <i>CPO</i> and user), average energy mix used |
| R24 | The system must be able to notify user of exception |
| R25 | The system must be able to notify user on successful action |
| R26 | The system must store the history of charging stations performance |
| R27 | The system must allow registered Driver to login |
| R28 | The system must allow registered Administrator to login |

3.2.1 Mapping on Goals

| Goal | Domain assumption | Requirement |
|------|-------------------------------------|---|
| G1 | D1, D2, D3, D5, D13 | R4, R5, R6, R8, R24 R25, R29 |
| G2 | D1, D2, D4, D10, D5, D13 | R4, R5, R8, R9, R16, R24, R23, R24, R27 |
| G3 | D7, D8, D9, D11, D12, D13, D14, D15 | R7, R9, R10, R22, R23, R24, R27 |
| G4 | D3, D6 | R17, R18, R19, R20, R21, R22, R25, R26, R28 |
| G5 | D3, D6 | R17, R18, R19, R20, R21, R22, R23, R24, R25, R26, R28 |
| G6 | D3, D6 | R17, R21, R22, R23, R24, R25, R26, R28 |
| G7 | D1, D2, D3, D5, D13 | R2, R14, R15, R16, R24, R25, R27 |

3.2.2 Use cases

1. Driver registration

| | |
|------------------|--|
| Actor | Driver |
| Entry conditions | The Driver does not have an account and is on the main page (initial view) of the system |
| Events flow | <ol style="list-style-type: none"> 1. The Driver clicks the “Sign Up” button 2. The Driver enters name, surname, a valid payment method email address and password and the licence plate of the first vehicle he wants to add 3. The Driver accepts the system to use the geolocalization service of his device and his daily schedule information 4. The Driver click on the submit button 5. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | New Driver account created |
| Exceptions | <ol style="list-style-type: none"> 1. The Driver does not enter all the mandatory data 2. The Driver does not enter a valid payment method 3. The Driver does not permit the system to access the localization service 4. The Driver enters a non-existing licence plate <ul style="list-style-type: none"> • In all cases <i>eMall</i> notifies the Driver displaying an error message |

2. User login to *eMall*

| | |
|------------------|--|
| Actor | Driver or Administrator |
| Entry conditions | The Driver or the Administrator is on the main page of the system |
| Events flow | <ol style="list-style-type: none">1. The Driver/Administrator clicks on the "Sign In"2. The Driver/Administrator enters email and password3. The Driver/Administrator clicks on the submit button4. <i>eMall</i> processes the information and redirects the Driver or the Administrator to the relative Stations Page |
| Exit conditions | The Driver or the Administrator logs in |
| Exceptions | <ol style="list-style-type: none">1. The Driver/Administrator does not enter a correct password for the email/authentication code2. The email does not exist in the system3. The Driver enters a non-existing licence plate<ul style="list-style-type: none">• In all cases <i>eMall</i> notifies the User displaying an error message |

3. Driver adds new vehicle

| | |
|------------------|---|
| Actor | Driver |
| Entry conditions | The Driver is logged into the system and is in Stations Page |
| Events flow | <ol style="list-style-type: none">1. The Driver clicks on UserProfile Page2. The Driver clicks on "Enter new vehicle"3. The Driver enters the licence plate of his vehicle.4. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | The Driver has a new vehicle available |
| Exceptions | <ol style="list-style-type: none">1. The Driver enters a licence plate that does not exist or is already exists in the database<ul style="list-style-type: none">• In this case <i>eMall</i> notifies the Driver displaying an error message |

4. Driver changes active vehicle

| | |
|------------------|--|
| Actor | Driver |
| Entry conditions | The Driver is logged into the system and is in Stations Page |
| Events flow | <ol style="list-style-type: none">1. The Driver clicks on UserProfile Page2. The Driver clicks on “Change Active Vehicle”3. The Driver chooses one vehicle from his vehicle list as active4. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | The chosen vehicle is set as the Active Vehicle |
| Exceptions | No exception |

5. Driver adds new payment method

| | |
|------------------|---|
| Actor | Driver |
| Entry conditions | The Driver is logged into the system and is in Stations Page |
| Events flow | <ol style="list-style-type: none">1. The Driver clicks on UserProfile Page2. The Driver clicks on “Add new payment method”3. The Driver fills the form with the required data4. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | The Driver has a new payment method available |
| Exceptions | <ol style="list-style-type: none">1. The Driver enters wrong payment data in the form<ul style="list-style-type: none">• In all cases <i>eMall</i> notifies the Driver displaying an error message |

6. Driver changes active payment method

| | |
|------------------|--|
| Actor | Driver |
| Entry conditions | The Driver is logged into the system and is in Stations Page |
| Events flow | <ol style="list-style-type: none">1. The Driver clicks on UserProfile Page2. The Driver clicks on “Change active payment method”3. The Driver chooses one vehicle from his payment method list as active4. <i>eMall</i> processes the information and shows a success message |

| | |
|-----------------|--|
| Exit conditions | The payment method is set as the Active Payment Method |
| Exceptions | No exception |

7. Driver makes a reservation

| | |
|------------------|---|
| Actor | Driver |
| Entry conditions | The Driver is logged into the system and is in Stations Page |
| Events flow | <ol style="list-style-type: none">1. The Driver visualises the stations from the sorted list or, clicking on Map Page, from the map view2. The Driver clicks on the station where he wants to book a socket for his vehicle3. The Driver selects the socket (slow, fast..) among the available4. The Driver selects the time of the charge5. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | The Driver has a new event in his calendar and in his MyReservations Page, with the relative reservation details (station, timeframe and socket number) |
| Exceptions | <ol style="list-style-type: none">1. The Driver clicks a station where there are no socket accepted by the Driver's Active Vehicle.<ul style="list-style-type: none">• In this case <i>eMall</i> notifies the Driver displaying an error message |

8. Driver cancels a reservation

| | |
|------------------|---|
| Actor | Driver |
| Entry conditions | The Driver is logged into the system and is in Stations Page and has at least a reservation |
| Events flow | <ol style="list-style-type: none">1. The Driver goes to MyReservations Page where a list of his reservation is displayed2. The driver clicks on the reservation he wants to cancel3. The Driver clicks on the "Cancel" button4. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | The reservation is removed |
| Exceptions | <ol style="list-style-type: none">1. The reservation is in less than 15 minutes |

| | |
|--|---|
| | <ul style="list-style-type: none">• In this case <i>eMall</i> notifies the Driver displaying an error message |
|--|---|

9. Driver starts the charging process

| | |
|------------------|--|
| Actor | Driver |
| Entry conditions | The Driver is logged in the system with at least an active reservation |
| Events flow | <ol style="list-style-type: none">1. The Driver clicks on MyReservations2. The driver clicks on the reservation3. The Driver clicks on “Start charging”4. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | The charging process starts |
| Exceptions | <ol style="list-style-type: none">1. The socket is not correctly plugged in2. There is not enough energy in the station <ul style="list-style-type: none">• In all cases <i>eMall</i> notifies the Driver displaying an error message |

10. Driver view the charging status

| | |
|------------------|--|
| Actor | Driver |
| Entry conditions | The Driver is logged into the system and is in Stations Page |
| Events flow | <ol style="list-style-type: none">1. The Driver clicks on MyReservations2. The Driver clicks on one of his reservations3. The Driver clicks on “Charging status”4. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | The Driver visualises the charging status |
| Exceptions | <ol style="list-style-type: none">1. The charge process has not started yet <ul style="list-style-type: none">• In all cases <i>eMall</i> notifies the Driver displaying an error message |

11. Driver reaches a station

| | |
|-------|--------|
| Actor | Driver |
|-------|--------|

| | |
|------------------|--|
| Entry conditions | The Driver is logged into the system and is in Stations Page |
| Events flow | <ol style="list-style-type: none">1. The Driver has 2 alternatives: he can click on MyReservations Page and then on the reservation or he can click on the station desired from the Stations Page2. The Driver clicks on “Reach charging station”3. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | Driving directions are shown on the navigation system on the Driver’s device |
| Exceptions | No exception |

12. Driver accepts a recommendation

| | |
|------------------|--|
| Actor | Driver |
| Entry conditions | The Driver is logged into the system and is in Stations Page |
| Events flow | <ol style="list-style-type: none">1. The Driver clicks on MyRecommendations Page2. The Driver clicks on one of the two recommendations displayed (one based on his calendar and the other on his vehicle battery status)3. The Driver clicks on “Accept Recommendations”3. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | A new reservation is created and added to Driver’s MyReservations Page and device calendar |
| Exceptions | No exception |

13. Administrator modify the energy mix

| | |
|------------------|---|
| Actor | Administrator |
| Entry conditions | The Administrator is logged into the system and is in Stations Page |
| Events flow | <ol style="list-style-type: none">1. The Administrator clicks on the desired station2. The Administrator clicks on StationSettings Page3. The Administrator manually modify the EnergyMix or sets the automatic change mode3. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | Station's mix of supply sources for vehicle charges changes |
| Exceptions | <p>There is not enough power to change the mix of supply sources, this could be due to lack of energy provided by the <i>DSO</i> or the energy in the stationary battery is not enough</p> <ul style="list-style-type: none">• In all cases <i>eMall</i> notifies the Driver displaying an error message |

14. Administrator modify the *DSO*

| | |
|------------------|---|
| Actor | Administrator |
| Entry conditions | The Administrator is logged into the system and is in Stations Page |
| Events flow | 1. The Administrator clicks on <i>DSO</i> Page 2. The Administrator chooses the <i>DSO</i> from the relative list, where they are visualised along with their energy price, or decides to set the automatic change mode 3. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | The <i>DSO</i> is changed |
| Exceptions | No exception |

15. Administrator set an offer

| | |
|------------------|---|
| Actor | Administrator |
| Entry conditions | The Administrator is logged into the system and is in Stations Page |
| Events flow | 1. The Administrator clicks on the station where he wants to set an offer 2. The Administrator goes to StationSettings Page 3. The Administrator clicks on “Set offer” 3. <i>eMall</i> processes the information and shows a success message |
| Exit conditions | The station has a new offer on its sockets |
| Exceptions | No exception |

16. Administrator view station information and performance

| | |
|------------------|--|
| Actor | Administrator |
| Entry conditions | The Administrator is logged into the system and is in Stations Page |
| Events flow | 1. The Administrator clicks on the station 2. <i>eMall</i> processes the information and shows the data of the station (energy stored in the battery, monthly performance, sockets available, etc...) |

| | |
|-----------------|---|
| Exit conditions | Station performance, statistics and information are shown |
| Exceptions | No exception |

3.2.3 Use case diagrams

1. Unregistered Driver



Fig. 6: *Unregistered Driver use case*

2. Administrator

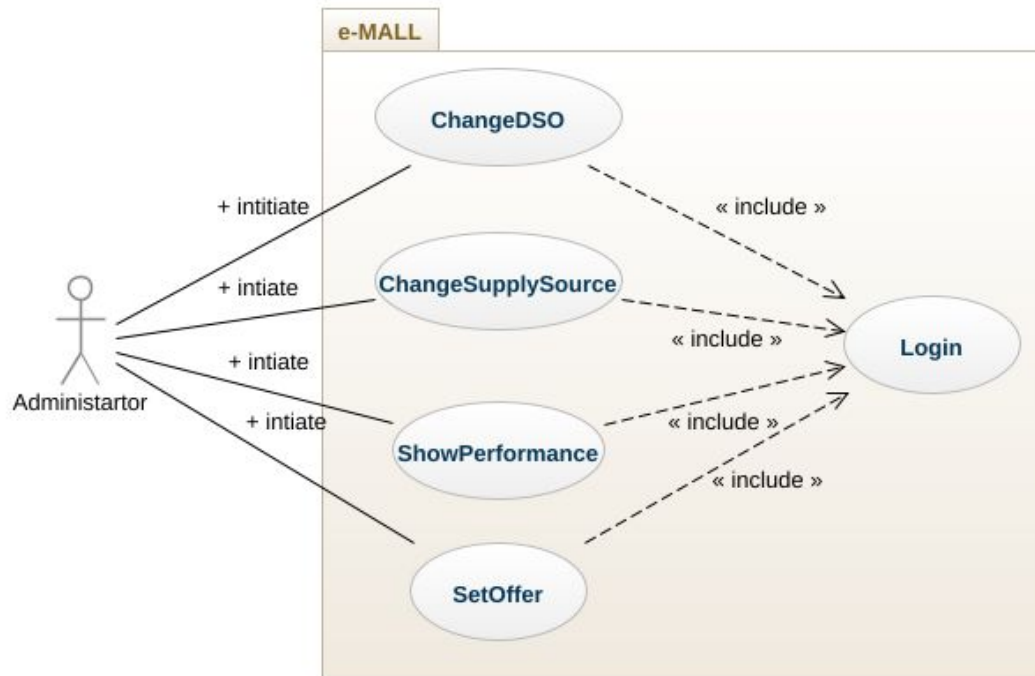


Fig. 7: Administrator use case

3. Driver

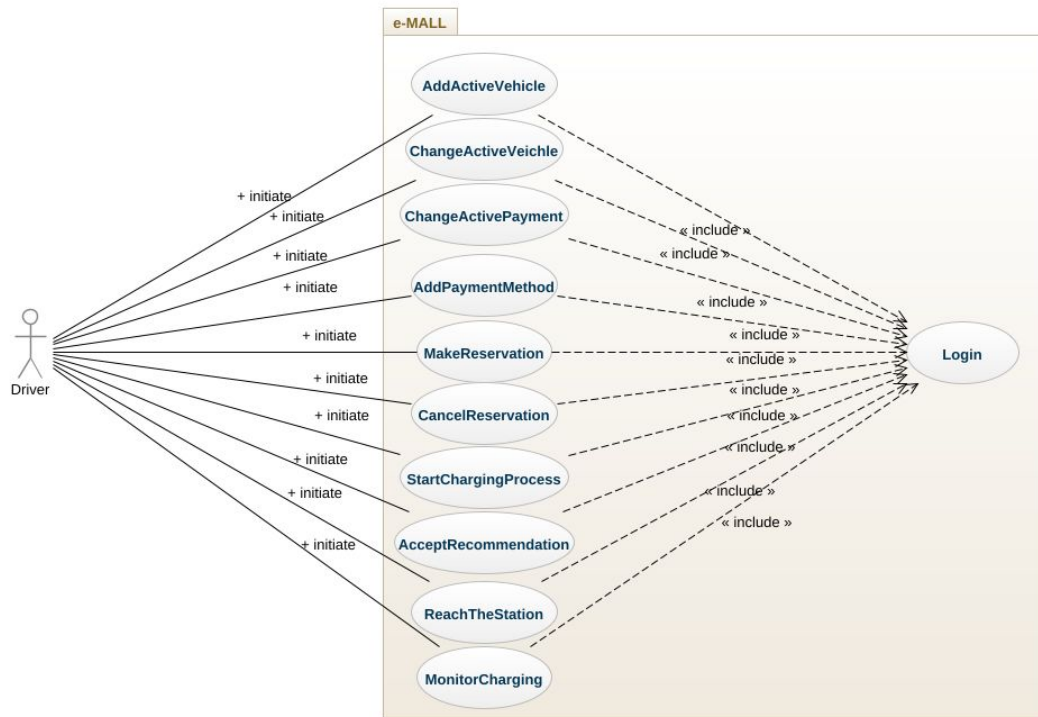


Fig. 8: Driver use case

3.2.4 Sequence diagram

1. Driver registration

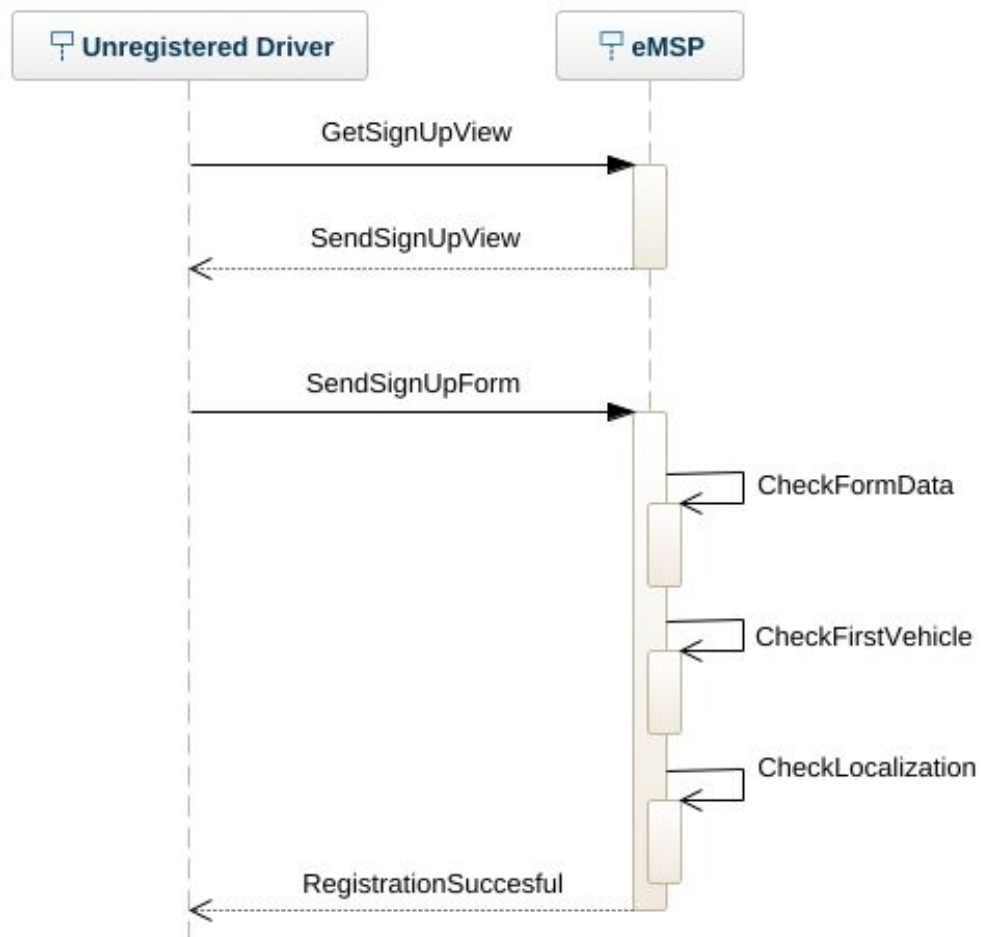
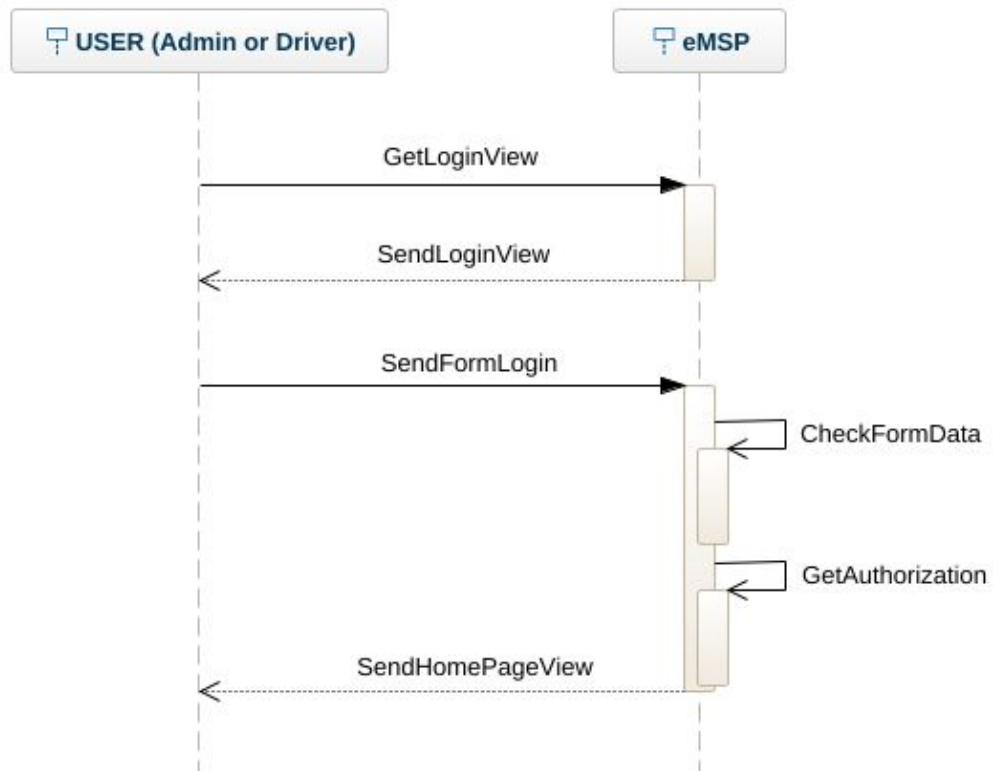


Fig. 9: Sequence diagram of the registration process for a Driver

After the Driver enters the Profile Information he will be asked to enter his vehicle information which will be set as his active vehicle. In **CheckFormData** is checked if all the information entered are correct. In **CheckFirstVehicle** it is checked if the licence plate exists. In **CheckLocalization** it is checked if the device GPS is available.

2. User login to *eMall***Fig. 10:** Sequence diagram of the login to eMall

3. Driver add a new vehicle

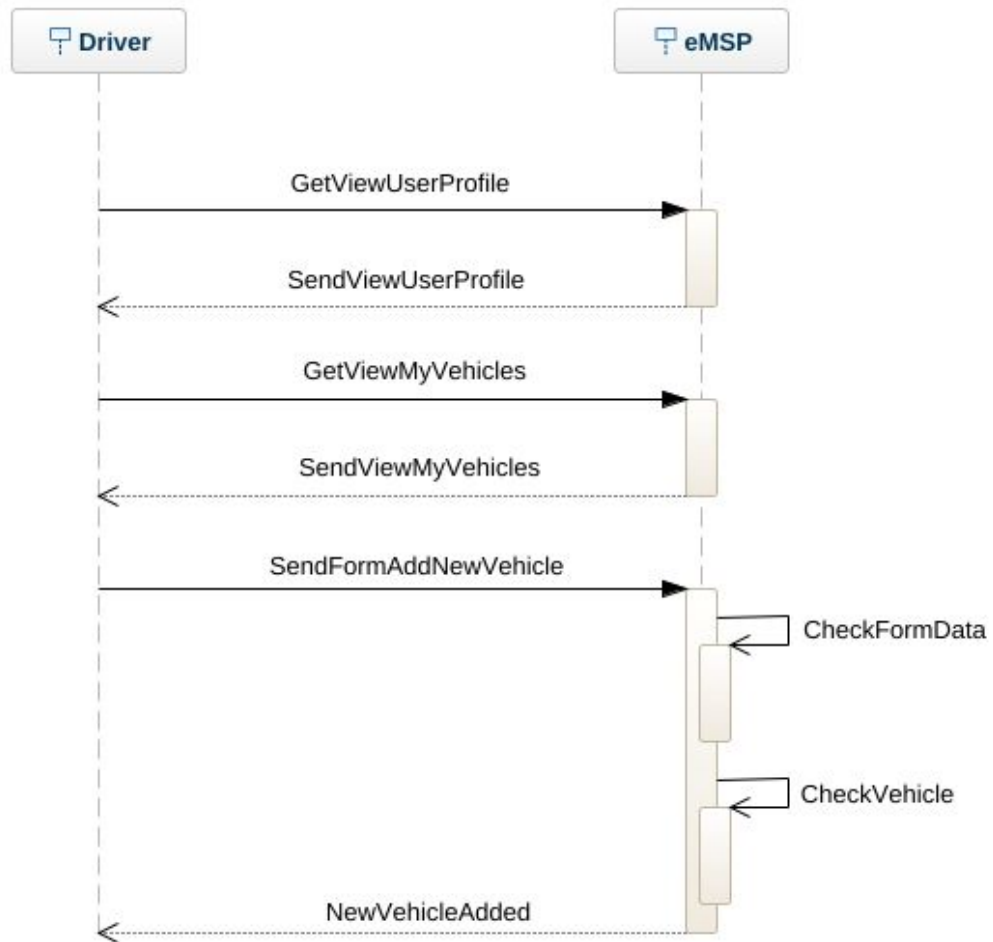
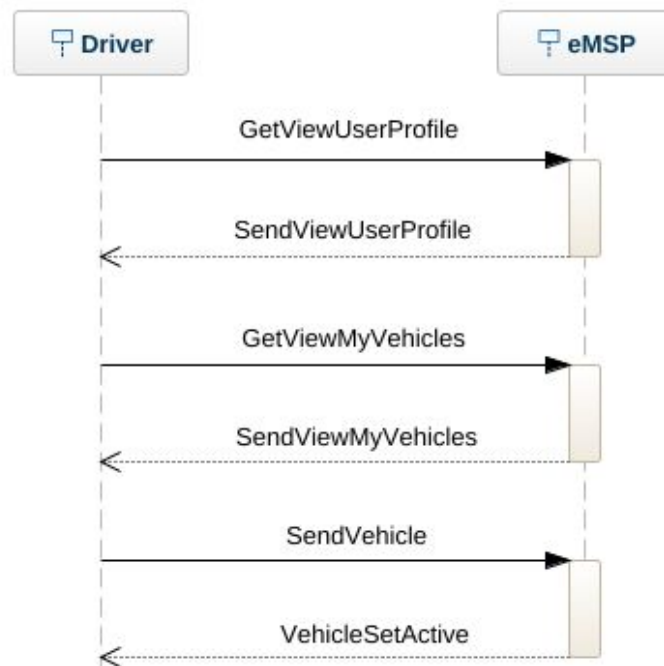


Fig. 11: Sequence diagram of a Driver adding a new EV to his profile

In `CheckFormData` is checked if the licence plate entered is correct. In `CheckVehicle` is checked if the vehicle is already present in the *eMall* database.

4. Driver change active vehicle

**Fig. 12:** Sequence diagram of a Driver changing the Active Vehicle

After clicking on "Change Active Vehicle" in the UserProfile Page, the Driver can select his current active Vehicle. The Active Vehicle will be used to show available stations and to link reservations to it. It will be also useful for the system to make custom recommendations.

5. Driver add new payment method

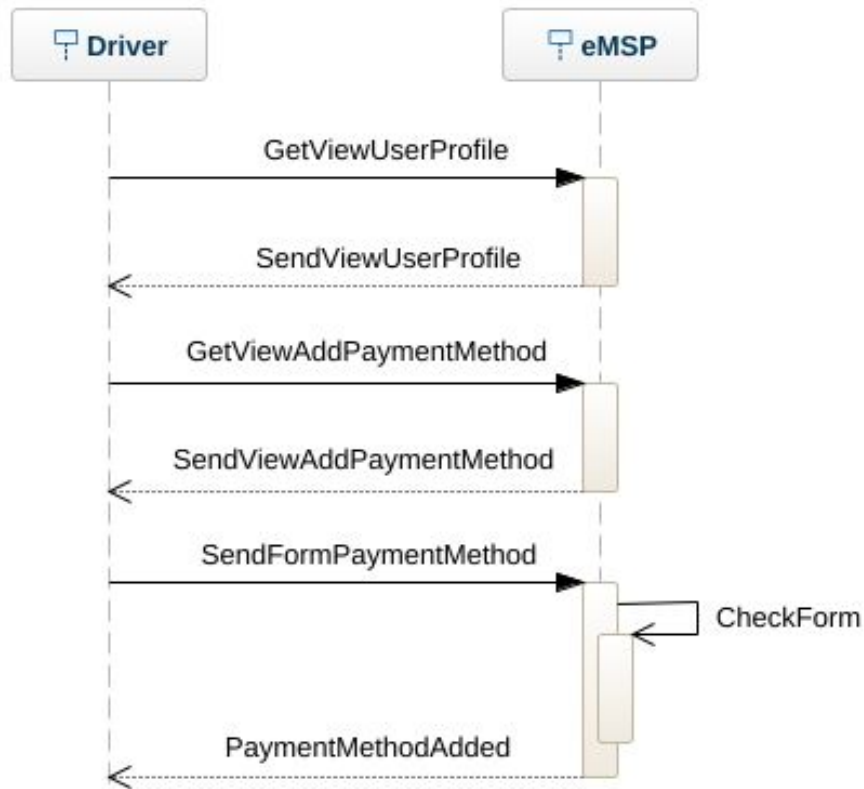


Fig. 13: Sequence diagram of a Driver adding a new payment method to his profile

The Driver clicks on the UserProfile Page and then on “Add payment method”. After that, the Driver has to fill the form with all the mandatory data about his payment method and submit it. The *eMSP* with *CheckForm* checks if the data is correct and notifies the Driver of the successful addition of the new payment method.

6. Driver change active payment method

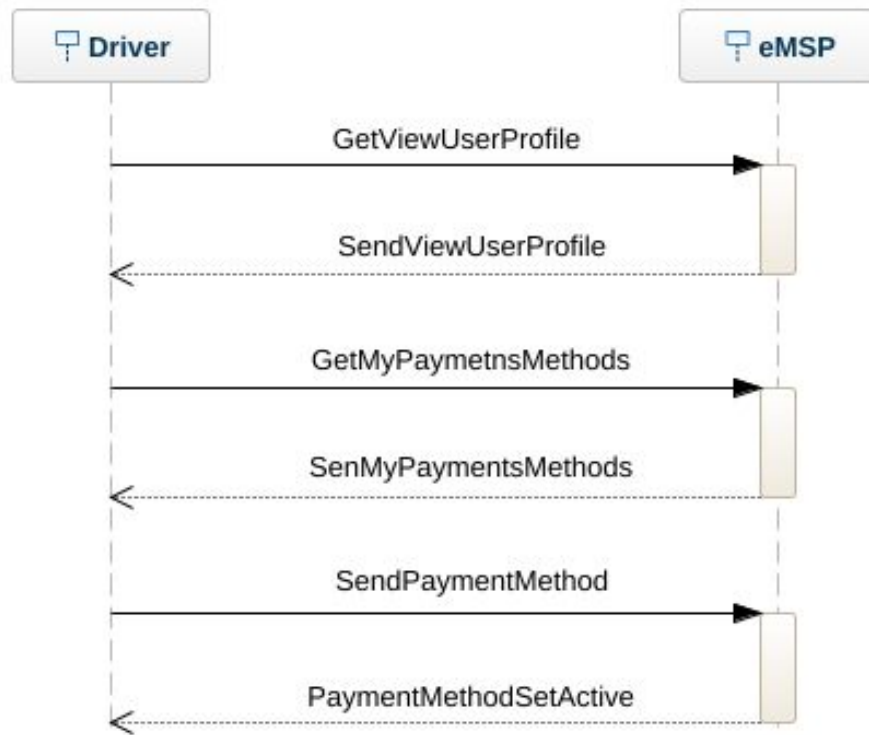


Fig. 14: Sequence diagram of a Driver changing the Active Payment Method

The Driver goes to the UserProfile Page and selects “Change active payment method”. A list of Driver’s payment methods is shown, afterwards the Driver chooses the payment method he wants to be active and the *eMSP* will proceed to change the Driver’s Active Payment Method.

7. Driver make a reservation

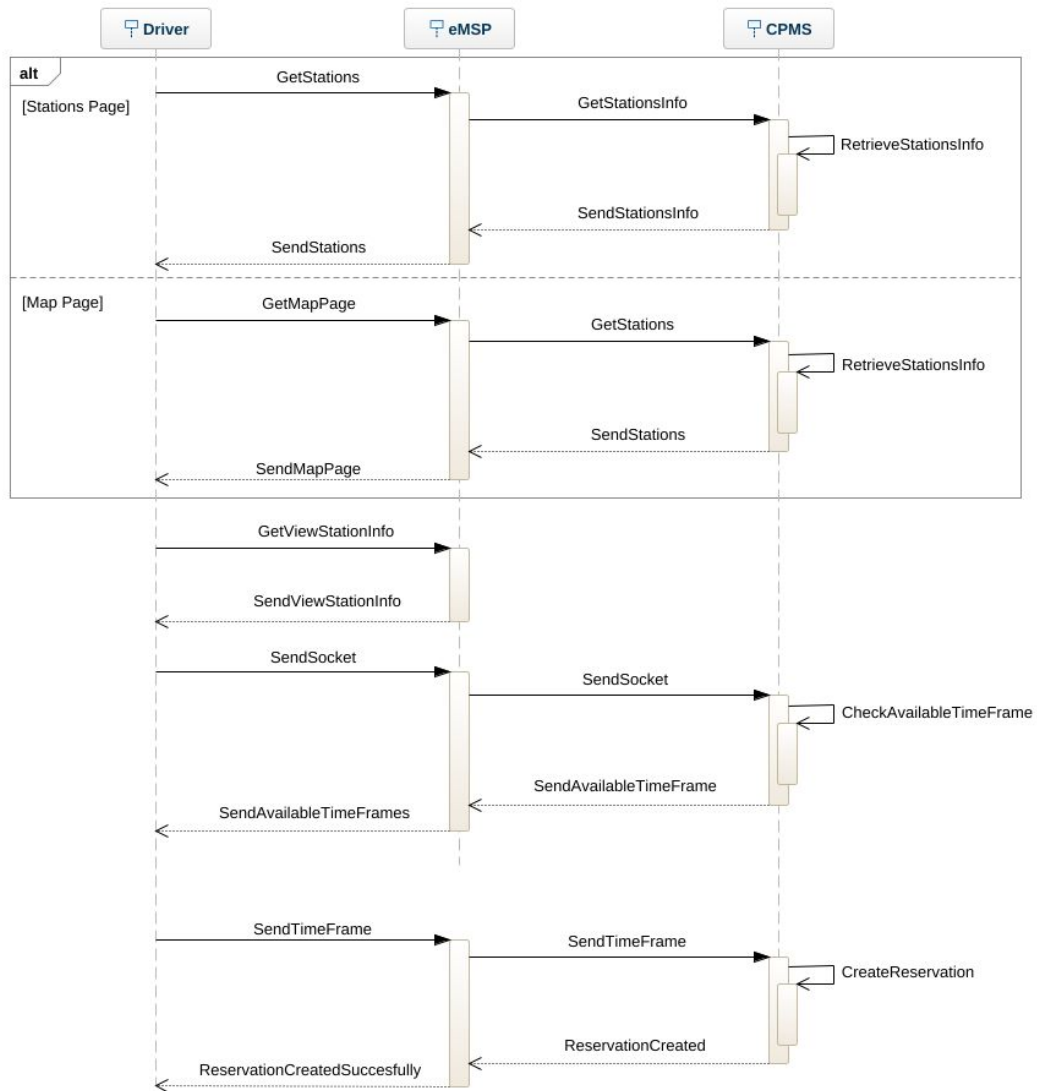


Fig. 15: Sequence diagram of a Driver making a new reservation

The two alternative sequences represent the possibility for the Driver to choose the desired station from the Map Page or from the Stations Page. The CPMS, with `RetrieveStationInfo`, retrieves all the information about the stations it manages. Then the Driver, after selecting the station, clicks on the available socket he wants to book. Afterwards, the system shows the time frames available for that

socket. The user selects the preferred timeframe and eventually the reservation is created.

8. Driver cancel a reservation

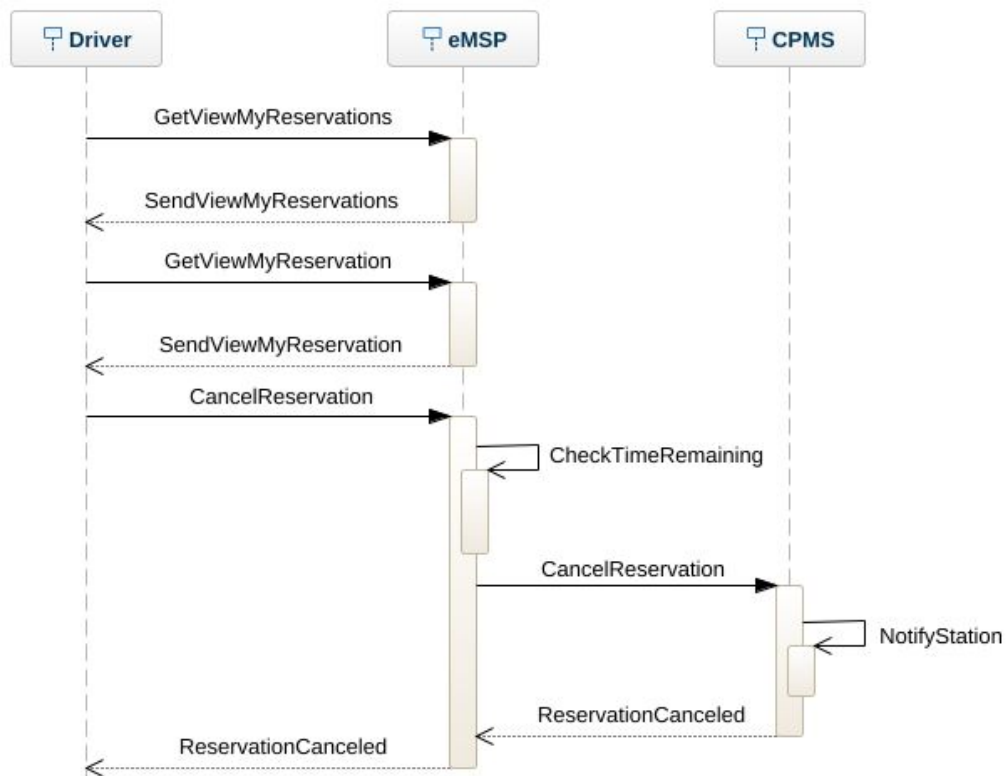


Fig. 16: Sequence diagram of a Driver canceling a reservation

The Driver goes to the MyReservations page where a list of the Driver's reservations is shown, then the Driver clicks on the "CANCEL" button on the reservation he wants to cancel. With CheckTimeRemaining, the *eMSP* checks if the reservation is in less than 15 minutes and eventually deletes it.

9. Driver start the charging process

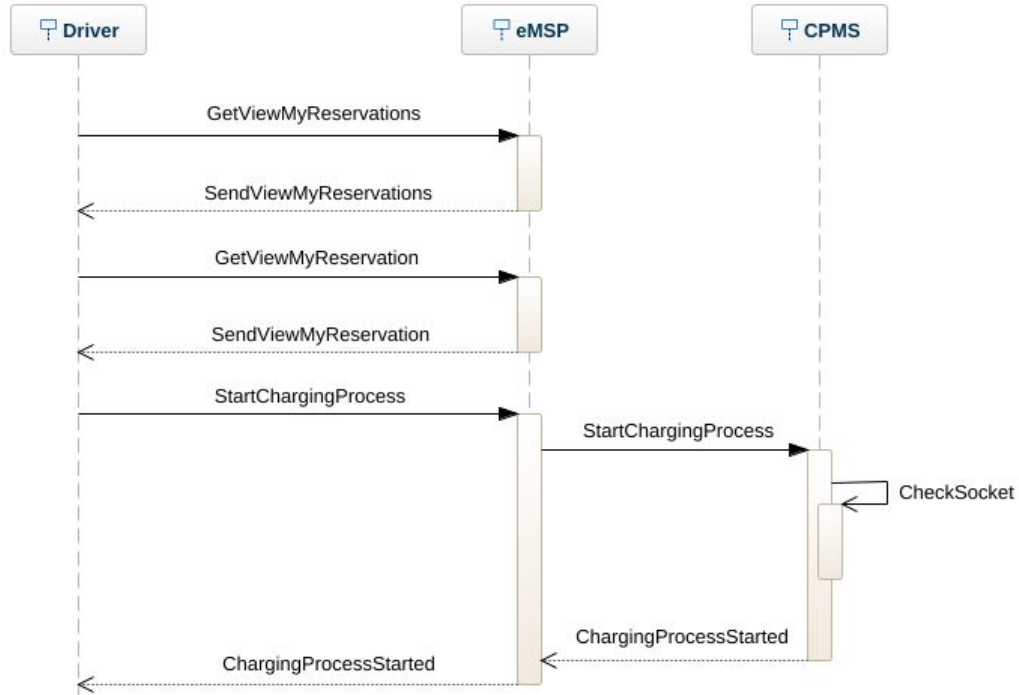


Fig. 17: Sequence diagram of a Driver starting the charging process

The *CPMS* receives the request and, with *CheckSocket*, verifies if the socket is correctly inserted and if there is energy enough in the station. If so, the station will notify the *CPMS* of the beginning of the recharging process.

10. Driver view charging status

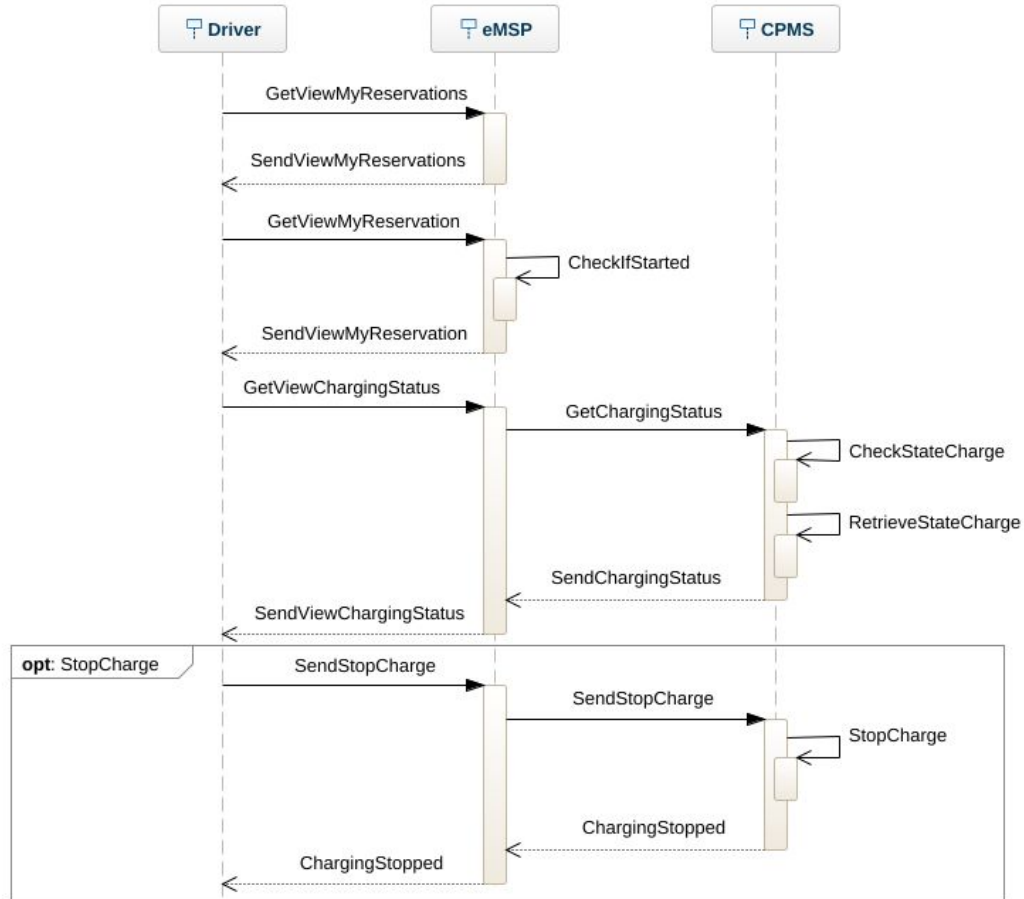


Fig. 18: Sequence diagram of a Driver monitoring the charging process

The Driver goes to MyReservation Page and clicks on the reservation. The *eMSP*, with `CheckIfStarted`, checks if the reservation has already started or is about to start. Then, the Driver clicks on “Charging status”. The *CPMS*, with `CheckStateCharge`, checks if the battery has been fully recharged and if so stops charging. The opt sequence represents the possibility for the Driver to manually stop the charging process.

11. Driver reach a station

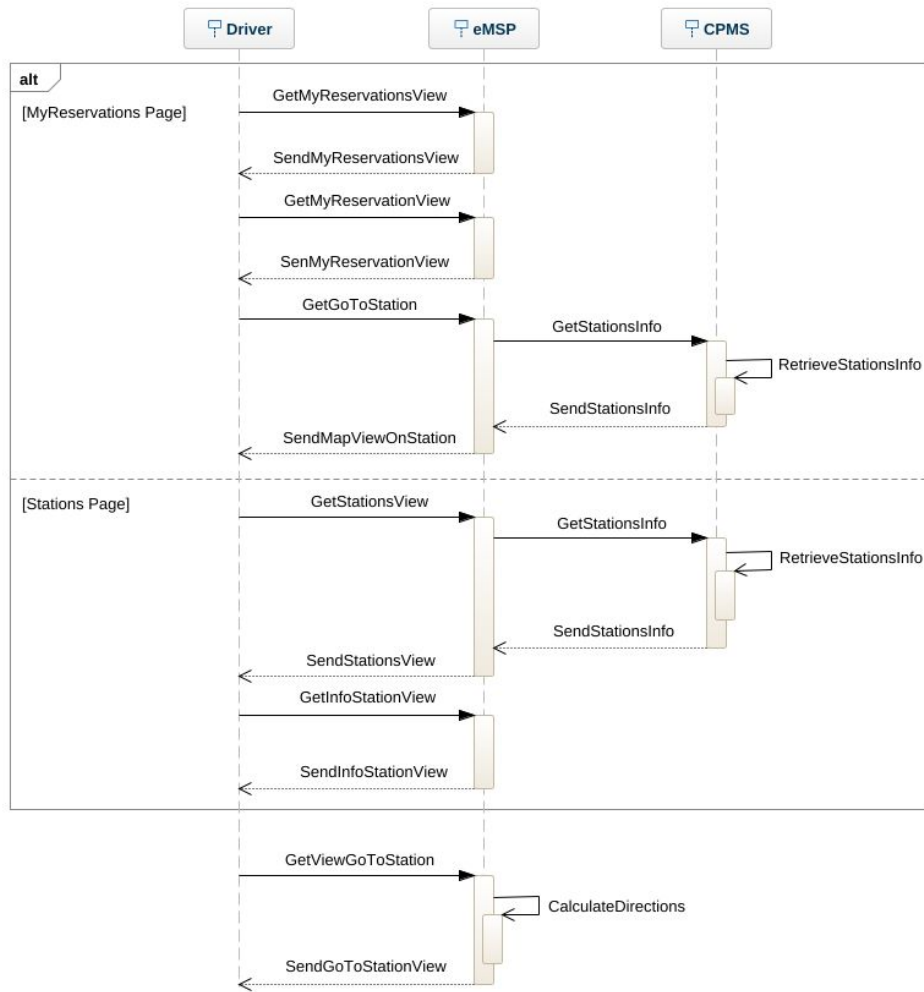


Fig. 19: Sequence diagram of a Driver that wants to reach a charging process

Here are shown the 2 different alternatives that the Driver can follow to get the directions for the desired station.

12. Administrator modify energy mix

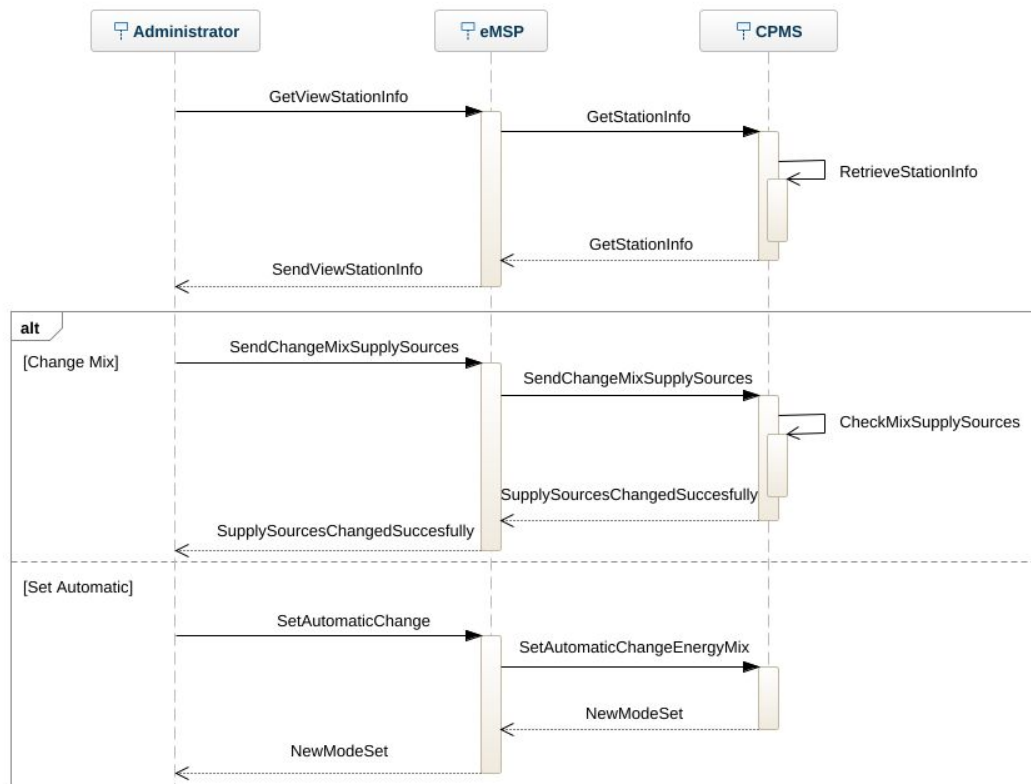


Fig. 20: Sequence diagram of an Administrator modifying the energy mix

The *CPMS* uses *RetrieveStationInfo* to retrieve all the data about a station. With *CheckMixSupplySources*, it is checked if there is enough energy to make the desired change. The Energy Mix can be manually set to the desired value or in alternative can be set to be automatically changed by the *CPMS*.

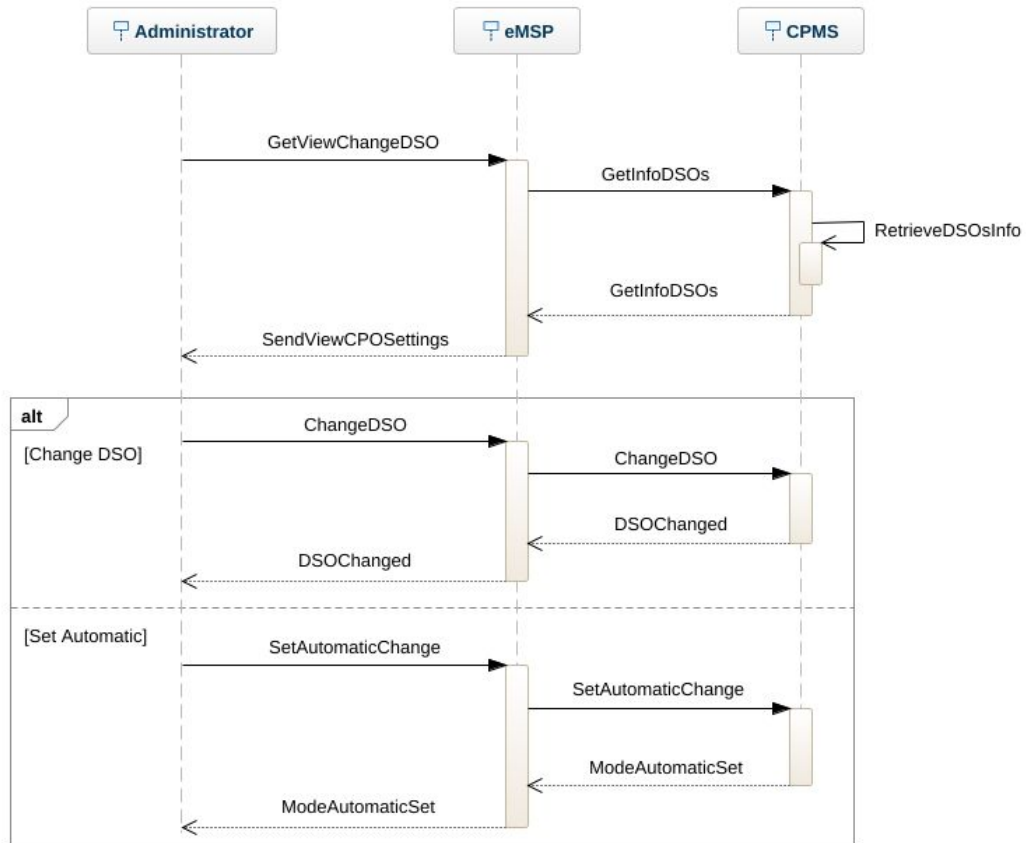
13. Administrator modify *DSO*

Fig. 21: Sequence diagram of an Administrator that modifies the *DSO*

With `RetrieveDSOsInfo` the *CPMS* will retrieve all the data about his *DSOs*. The *DSO* can be manually selected or in alternative can be set to be automatically selected by the *CPMS*.

14. Administrator set an offer

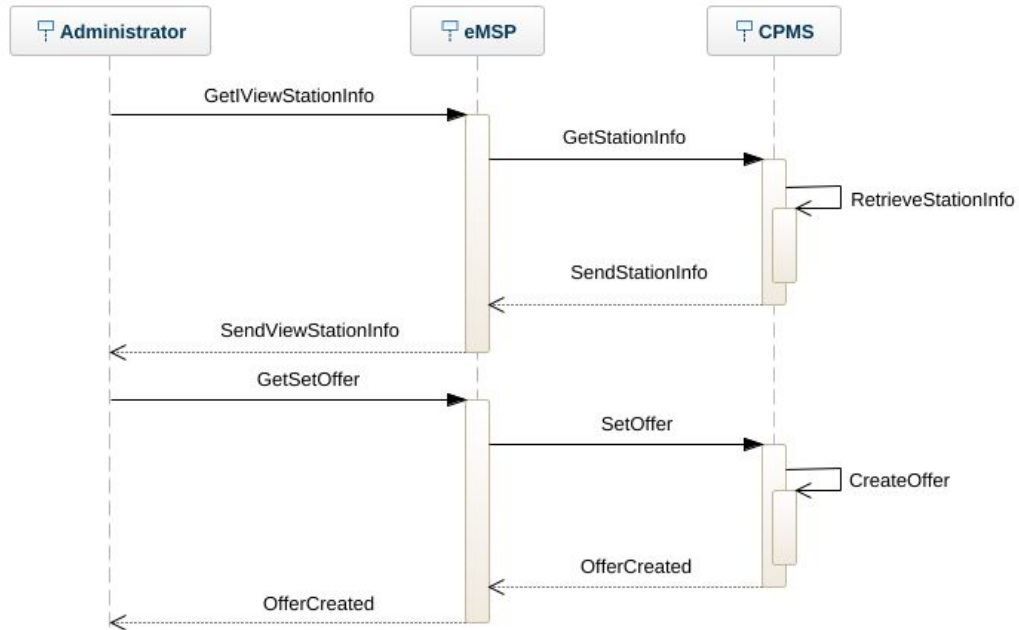


Fig. 22: Sequence diagram of an Administrator setting an offer

The Administrator, after selecting a station, clicks on "Charging Options" and then can either choose to modify the current price or to add a new special offer, for a certain socket type. With `CreateOffer` the *CPMS* will add the offer to the desired station.

15. Administrator view station information

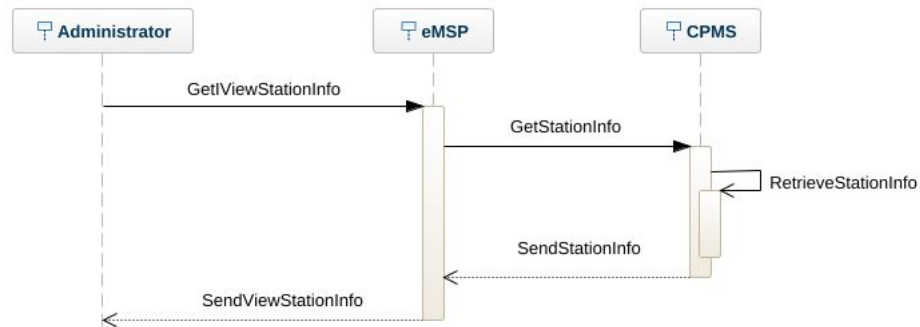


Fig. 23: Sequence diagram of an Administrator checking stations performance

The Administrator selects the desired station from the list in Stations Page. The CPMS retrieves the data with RetrieveStationInfo and then, the system shows the view with the station information, statistics and performance to the Administrator.

16. Driver Accept a recommendation

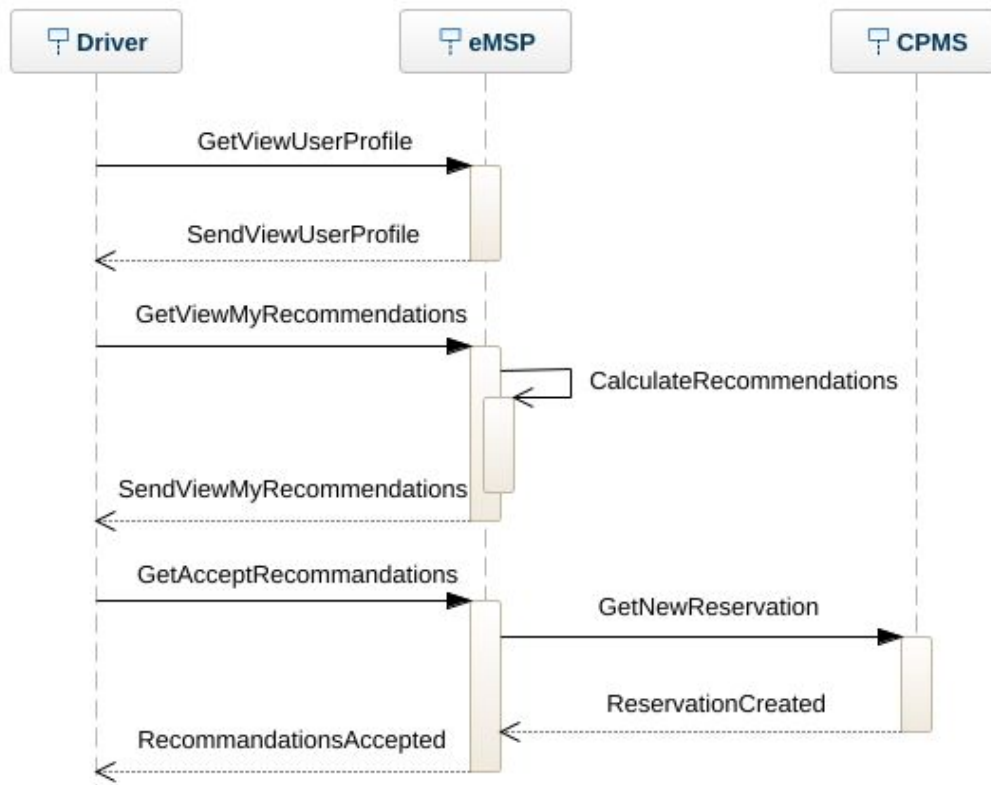


Fig. 24: Sequence diagram of a Driver accepting a recommendation

The Driver goes on the UserProfile Page and clicks on “My Recommendations”. The *eMSP* handles the request and retrieves the recommendations made for the Driver. After the Driver clicks on “Accept Recommendation”, the new reservation is created and inserted in the database.

3.2.5 Mapping on requirements

| Use case | Requirements |
|--|--|
| Driver registration | R1, R2, R24, R25 |
| User login to <i>eMall</i> | R24, R25, R27, R28 |
| Driver add new vehicle | R2, R24, R25, R27 |
| Driver change active vehicle | R2, R24, R25, R27 |
| Driver add new payment method | R3, R24, R25, R27 |
| Driver change active payment method | R3, R24, R25, R27 |
| Driver make a reservation | R4, R5, R6, R7, R8, R24, R25, R27 |
| Driver cancel a reservation | R9, R16, R24, R25, R27 |
| Driver start the charging process | R9, R11, R24, R25, R27 |
| Driver view charging status | R9, R12, R13, R24, R25, R27 |
| Driver reach a station | R4, R5, R6, R10, R24, R25, R27 |
| Administrator modify energy mix | R17, R20, R21, R22, R23, R24, R25, R28 |
| Administrator modify the <i>DSO</i> | R21, R22, R24, R25, R28 |
| Administrator set an offer | R17, R18, R22, R23, R24, R25, R28 |
| Administrator view station information | R17, R18, R19, R20, R23, R24, R25, R28 |
| Driver accept a recommendation | R14, R15, R24, R25, R27 |

3.3 Performance Requirements

The system must be reliable to handle an eventual incorrect procedure and must be supported by a fast internet infrastructure that is fundamental to guarantee to the users the possibility to evaluate only the charging stations actually available, without the risk of trying to make a reservation on a socket already booked. It is also extremely important to enable proper recommendations, based on information mostly retrieved by external APIs. This performance goal could be reached relying on an internet provider that ensures the user capacity and speed required.

Secondly, the system should be able to handle many concurrent users, at least 10 000, and many concurrent inputs.

3.4 Design Constraints

3.4.1 Standard compliance

All user data should be treated in compliance with GDPR (or the local privacy law), the authority that defines how the companies that work in the EU should collect, store and handle users' personal data. Moreover, the application should function fully on all

widely used web browsers and mobile stores. Lastly, the regulation and guidelines of external APIs must be followed.

3.4.2 Hardware limitations

This system will be made available both as a website and as a mobile application. One of the requirements the hardware has to satisfy is the possibility to access the internet through a web browser (valid method for smartphone or PC) rather than to download the application from the online store for mobiles. Moreover, the hardware of the Driver must have the possibility to create schedules according to the user's duties and his vehicles must be provided with a localization system.

3.5 Software System Attributes

eMall system has to be available 24h/7d, without considering the scheduled maintenance breaks during the year, and both mobile application and website must be reactive and usable.

3.5.1 Reliability

The service has to guarantee a high reliability, so the system has to be available for an adequate period without interruptions. Since the system reliability depends on its subsystems components, both *eMSP* and *CPMS* must respect these parameters. The reliability of external APIs is not considered in this document and is out of system's control. To guarantee this high reliability, there should be scheduled and periodical maintenance interventions to prevent unexpected downtimes. Additionally, a duplicate of the server should be run in parallel to guarantee the service in case of failure. There should also be a backup of each database, *eMSP* and *CPMS*.

3.5.2 Availability

Since *eMall* service is based on the concept that each user should find the best charging option for his vehicle according to his temporary conditions (schedule, localization, etc. . .) in the minimum time possible, the system's availability should be as high as pos-

sible, in particular during the most crowded hours. It is supposed to have an availability of 99.9%, which means 9 hours/year of downtime. This goal can be reached handling the complexity of the individual components, having a high quality maintenance and duplicating server and databases, as previously explained in relation to reliability.

3.5.3 Security

The system must ensure a high standard of security since it stores and handles private and critical users' information, such as password, localization and credit card details. To achieve this goal, critical data should be saved on a DB hashed and salted and every input and request by the user must be sanitised. Moreover, there should be an authorization check at every API endpoint within the system.

3.5.4 Maintainability

The system should be easy to maintain and to be modified to correct faults, improve performance or to be adapted to a changed environment. To achieve this goal, the system should be based on one or more software patterns, which (possibly supported by a clear documentation) guarantee and facilitate future extensions of the system, with the possibility to implement new features without the necessity of rethinking the whole concept.

3.5.5 Portability

Since the system can be supported both by mobile application and website, it should be properly designed to be visualised and to preserve its functionalities regardless of the device used (PC, different smartphone models).

4 Formal Analysis

4.1 Alloy Code

The formal analysis of the system is represented through the use of Alloy code in order to verify the consistency of the model presented and the needed constraints.

SIGNATURES

```
// Ids, integer values and dates will not be considered for simplicity
```

```
open util/boolean
```

```
sig Position {}
sig Recommendation{}
```

```
sig CPO {
  //cpoId: one CPOId
  availableDSO: some DSO,
  activeDSO: one DSO
}
```

```
sig DSO {
  //dsoId: one DSOId,
  //energyPrice: one Int
}
```

```
abstract sig SocketType{}
one sig SLOW extends SocketType {}
one sig FAST extends SocketType {}
one sig RAPID extends SocketType {}
```

```
abstract sig SocketStatus{}
one sig AVAILABLE extends SocketStatus {}
one sig UNAVAILABLE extends SocketStatus {}
one sig CHARGING extends SocketStatus {}
```

```
abstract sig RechargeStatus{}
one sig INPROGRESS extends RechargeStatus {}
one sig PAYED extends RechargeStatus {}
```

```
abstract sig User {
  //userId: one UserId
  //name: one String,
  //surname: one String,
```

```
    //documentId: one DocumentId,  
    //email: one Email,  
    //password: one String  
}  
  
sig CPOAdministrator extends User{  
    cpo: one CPO  
}  
  
sig EVDriver extends User {  
    paymentMethods: some PaymentMethod ,  
    vehicles: some Vehicle ,  
    recommendation: one Recommendation  
}  
  
sig Vehicle {  
    //licensePlate: one String,  
    //battery: one Int,  
    type: one SocketType ,  
    isActive: one Bool  
}  
  
sig Socket {  
    //socketId: one socketId,  
    type: one SocketType ,  
    active: one Bool ,  
    status: one SocketStatus  
}  
  
sig ChargingStation {  
    //stationId: one StationId,  
    //energyMix: one Int,  
    //price: one Int,  
    //battery: one Int,  
    position: one Position ,  
    cpo: one CPO,  
    sockets: some Socket ,  
    options: some ChargingOption  
}  
  
sig PaymentMethod {  
    //cardNumber: one String  
    isActive: one Bool  
}
```

```
sig Payment {  
  //paymentId: one PaymentId  
  //amount: one Int,  
  paymentMethod: one PaymentMethod  
}
```

```
sig Recharge {  
  payment: lone Payment,  
  status: one RechargeStatus,  
}
```

```
sig Reservation {  
  //reservationId: ReservationId  
  //fromDate: one Date,  
  //toDate: one Date,  
  socket: one Socket,  
  evdriver: one EVDriver,  
  vehicle: one Vehicle,  
  recharge: lone Recharge  
}
```

```
sig ChargingOption {  
  //cost: one Float,  
  type: one SocketType,  
  isOffer: one Bool  
}
```

FACTS

```
fact uniqueSocketsOfChargingStations {  
  no disjoint c1, c2: ChargingStation | some s: Socket |  
    (s in c1.sockets) && (s in c2.sockets)  
}
```

```
fact uniqueRechargeForReservations {  
  no disjoint r1, r2: Reservation | some c: Recharge |  
    (c in r1.recharge) && (c in r2.recharge)  
}
```

```
fact uniqueVehiclesOfEVDrivers {  
  no disjoint d1, d2: EVDriver | some v: Vehicle |  
    (v in d1.vehicles) && (v in d2.vehicles)  
}
```

```
fact uniquePaymentsInRecharges {
  no disjoint r1, r2: Recharge | r1.payment = r2.payment
}

fact uniquePositions {
  no disjoint s1, s2: ChargingStation | s1.position = s2.position
}

fact uniquePaymentMethodsOfEVDivers {
  no disjoint d1, d2: EVDriver | some p: PaymentMethod |
    (p in d1.paymentMethods) && (p in d2.paymentMethods)
}

fact uniqueChargingOptions {
  no disjoint c1, c2: ChargingStation | one o: ChargingOption |
    (o in c1.options) && (o in c2.options)
}

fact uniqueChargingOptionsInChargingStations {
  all disjoint o1, o2: ChargingOption, c: ChargingStation |
    (o1 in c.options && o2 in c.options) implies
    (o1.type != o2.type or o1.isOffer != o2.isOffer)
}

fact uniqueRecommendations{
  no disjoint d1,d2: EVDriver | d1.recommendation = d2.recommendation
}

fact allCPOHasAdmininistrator {
  all c: CPO | some a: CPOAdministrator | a.cpo = c
}

// All EVDrivers has at least one active vehicle in his vehicles
fact allEVDRiverHasVehicle {
  all u: EVDriver | one v: Vehicle |
    v in u.vehicles && v.isActive = True
}

// All EVDrivers has at least one active payment method
fact allEVDRiverHasPaymentMethod {
  all u: EVDriver | one p: PaymentMethod |
    p in u.paymentMethods && p.isActive = True
}
```

```

// CPOs exists only if they own one or more charging stations
fact allCPOHasChargingStation{
    all c: CPO | some s: ChargingStation | c = s.cpo
}

// All CPOs has at least one active DSO in his availableDSO
fact allCPOhasDSO {
    all c: CPO | one d: DSO | d = c.activeDSO && d in c.availableDSO
}

// Reservation associated to vehicle owned by the driver
fact onlyVehicleOfDriverInHisReservation{
    all r: Reservation, d: EVDriver, v: Vehicle |
        (d = r.evdriver && v = r.vehicle) implies v in d.vehicles
}

// Reservations for a socket can be made only by drivers who owns
// a vehicle of compatible recharging speed
fact allVehicleCompatibleWithSocket{
    all r: Reservation | one v: Vehicle | v = r.vehicle &&
        (v.type = SLOW implies r.socket.type = SLOW) &&
        (v.type = FAST implies r.socket.type = SLOW or r.socket.type = FAST)
}

// All available sockets have an available ChargingOption
// associated to the station
fact onlyChargingOptionsForAvailableSockets{
    all c: ChargingStation, s: Socket |
        s in c.sockets implies (one o: ChargingOption |
            o.type = s.type && o in c.options && o.isOffer = False)
}

// All ChargingOptions are present only if there is
// a corresponding socket of the same speed
fact allChargingOptionHasSocket{
    all c: ChargingStation, o: ChargingOption | o in c.options implies
        (one s: Socket | o.type = s.type && s in c.sockets)
}

// Recharge status is payed only if associated with a payment
fact rechargeStatusPayed {
    all r: Recharge | r.status = PAYED iff
        (one p: Payment | p in r.payment)
}

```



```

// Payments associated to correct PaymentMethod
fact allPaymentAssociatedToDriver {
  all p: Payment | one d: EVDriver, s: Reservation, r: Recharge |
    d = s.evdriver && p.paymentMethod in d.paymentMethods &&
    p = r.payment && r = s.recharge
}

// Recharge is in progress only if the socket is in charging status
fact allRechargeInProgressIfSocketInCharging{
  all r: Recharge, t: Reservation, s: Socket |
    r = t.recharge && s = t.socket &&
    r.status = INPROGRESS implies s.status = CHARGING
  all s: Socket | s.status = CHARGING implies
    (one r: Recharge, t: Reservation |
      r = t.recharge && s = t.socket && r.status = INPROGRESS)
}

// Two vehicles can not be recharged at the same time
fact uniqueRechargeForEVDriver{
  no disjoint r1, r2: Reservation |
    r1.evdriver = r2.evdriver && r1.recharge.status = INPROGRESS &&
    r2.recharge.status = INPROGRESS
}

fact allChargingVehiclesActive {
  all r: Recharge, s: Reservation |
    r = s.recharge && r.status = INPROGRESS implies
      s.vehicle.isActive = True
}

fact allResevationForAvailableSocket {
  all r: Reservation, s: Socket |
    s = r.socket implies s.status != UNAVAILABLE
}

fact allSocketsConnected {
  all s: Socket | one c: ChargingStation | s in c.sockets
}

fact allPaymentMethodsConnected {
  all p : PaymentMethod | one e: EVDriver | p in e.paymentMethods
}

```

```
fact allPaymentsConnected {
  all p : PaymentMethod | one e: EVDriver | p in e.paymentMethods
}

fact allDSOConnected {
  all d: DSO | some c: CPO | d in c.availableDSO
}

fact allVehiclesConnected {
  all v: Vehicle | one d: EVDriver | v in d.vehicles
}

fact allChargingOptionsConnected {
  all c: ChargingOption | one s: ChargingStation | c in s.options
}

fact allRechargesConnected {
  all r: Recharge | one s: Reservation | r in s.recharge
}

fact allRecommendationsConnected {
  all r: Recommendation | one d: EVDriver | r in d.recommendation
}

fact allPositionsConnected {
  all p: Position | one c: ChargingStation | p in c.position
}
```

- DYNAMIC MODELING -

```
pred createReservation [d: EVDriver, v: Vehicle, s: Socket, r: Reservation] {
  r.evdriver = d
  r.vehicle = v
  r.socket = s
}

pred addVehicle [d, d': EVDriver, v: Vehicle] {
  d'.vehicles = d.vehicles + v
}

pred addSocket [c, c': ChargingStation, s: Socket] {
  c'.sockets = c.sockets + s
}
```

```
run createReservation
```

```
pred world1 {  
    #EVDriver = 2  
    #ChargingStation = 1  
    some s: Socket | s.status = AVAILABLE  
    some s: Socket | s.status = CHARGING  
    #Payment >= 1  
    #Vehicle >= 3  
    #Reservation >=2  
    some c: ChargingOption | c.isOffer = True  
}  
  
pred world2 {  
    #DSO>=5  
    #ChargingStation >= 5  
    #EVDriver = 0  
}  
  
run world1 for 5  
  
run world2 for 5
```

4.1.1 First model

In the first model, corresponding to predicate world1, we focus on the relationships between evdrivers, vehicles, sockets, reservations, recharges and payments. With this module we want to underline the following:

- Every Driver has at least an active payment method and an active vehicle.
- Every recharge, if not in progress, is linked to a unique payment associated to a payment method of the corresponding Driver.
- Every vehicle is associated to a reservation only if it is compatible with the charging speed offered by the chosen socket.
- Charging Stations can provide special offers through additional charging options.

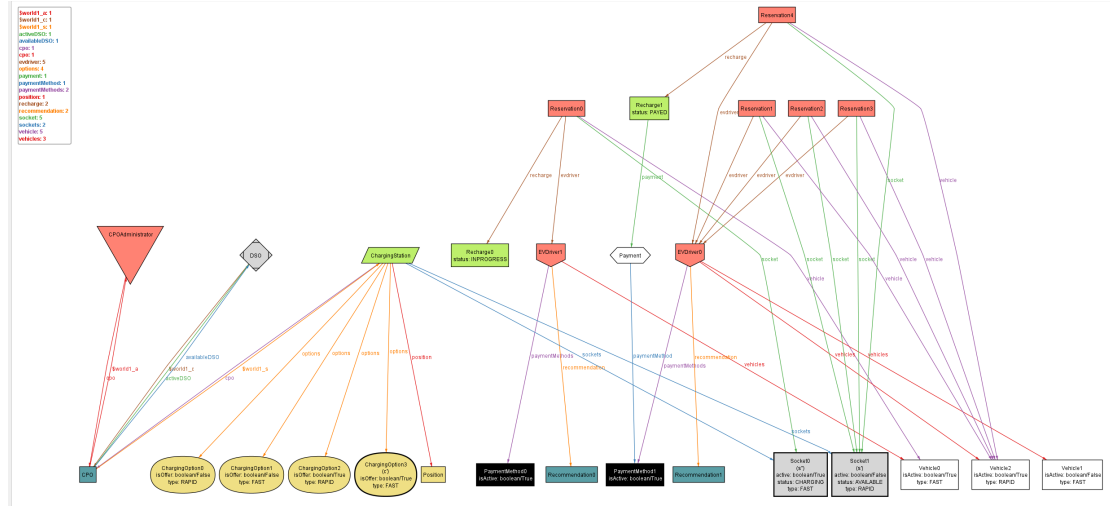


Fig. 25: A world obtained from the first model

4.1.2 Second model

In the second model, corresponding to predicate world2, we focus on the relationships between CPO administrators, dsos, charging stations and charging options. With this module we want to underline the following:

- Every CPO has only an active DSO between all available ones while multiple CPOs can have the same active DSO.
- Every charging station always offers a standard charging option for every type of socket

connected to it.

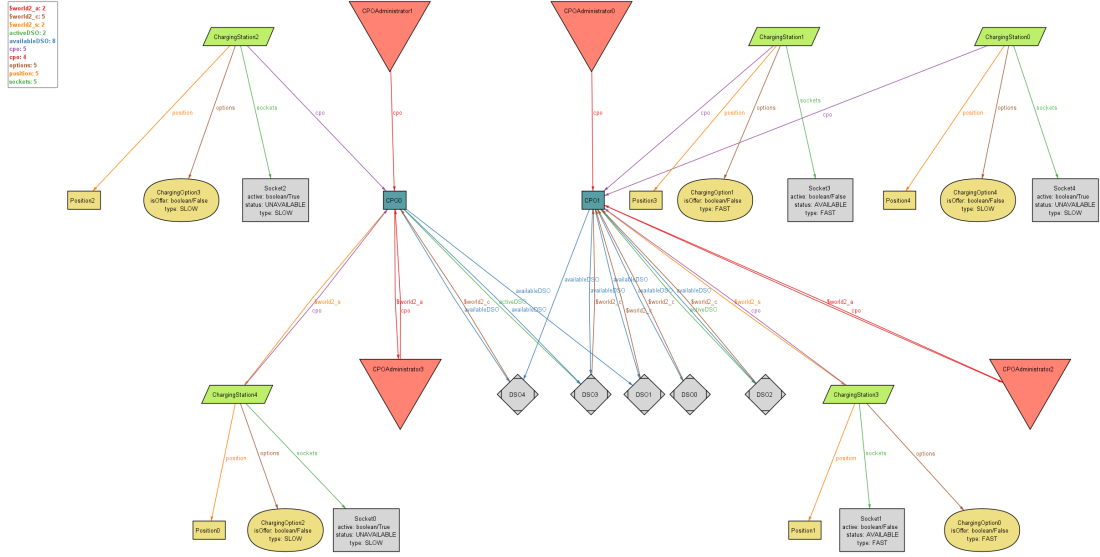


Fig. 26: A world obtained from the second model

4.1.3 Dynamic model

Some dynamic behaviour of the system have also been modelled:

- Creation of a reservation.
- Addition of a vehicle for an evdriver.
- Addition of a socket for a charging station.

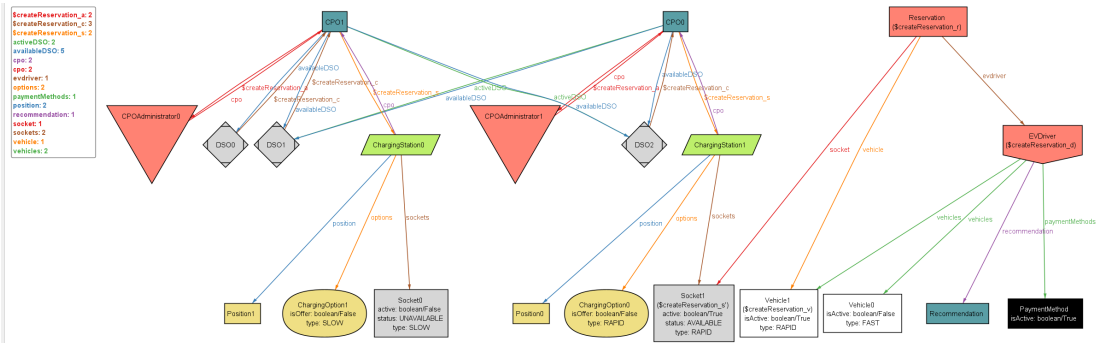


Fig. 27: A world obtained from the dynamic model

5 Effort spent

5.0.1 Roberto Cialini

| Section | Time spent |
|-----------------------|------------|
| Introduction | 6 |
| Overall description | 9 |
| Specific requirements | 6 |
| Formal analysis | 16 |
| Reasoning | 14 |
| Total time | 51 |

5.0.2 Umberto Colangelo

| Section | Time spent |
|-----------------------|------------|
| Introduction | 6 |
| Overall description | 10 |
| Specific requirements | 15 |
| Formal analysis | 7 |
| Reasoning | 14 |
| Total time | 52 |

5.0.3 Vittorio La Ferla

| Section | Time spent |
|-----------------------|------------|
| Introduction | 5 |
| Overall description | 5 |
| Specific requirements | 17 |
| Formal analysis | 8 |
| Reasoning | 14 |
| Total time | 49 |

6 References