POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

eMall

RASD Delivery (v1.0)

Author(s): **Tommaso Bonetti**
Fabio Ciani
Davide Mozzi

Reference professor: Elisabetta Di Nitto
Academic year: 2023-24
Release date: December 23, 2022

Contents

1 | Introduction

Electric mobility (e-Mobility) aims at providing sustainable transport to its customers, mainly to accomplish a limitation of the carbon footprints caused by humans in urban and sub-urban settings.

eMall (e-Mobility for all) is an ambitious project which strives for a standardization and unified connection between several companies in the sustainable industry, called charging point operators (CPOs). For this reason, a consortium has been established, and all the compliant business organizations abide by its guidelines. The head of the board represents the major client of eMall, who is asking the software engineering team to create an application which will implement both an **e-Mobility service provider** (eMSP) and a **charge point management system** (CPMS).

The CPMS subsystem of eMall, as consolidated in the smart energy and trasport sector, should also be able to communicate with energy vendors and suppliers, called distribution system operators (DSOs).

1.1 Purpose

The software must achieve the following results.

- Let the compliant companies seamlessly integrate proprietary and/or already present CPMSs on the market with eMall's eMSP platform in a software as a service (SaaS) approach, in order to exploit collaboration and share of information.
- Enable eMall's eMSP and CPMS services to retrieve necessary data from the configured external CPMSs and the DSOs, respectively.
- Offer the end users the possibility to take advantage of the eMSP features of the eMall application.

It should be noted that a few discussion involving the supporting organizations are not to be included in the scope of the analysis.

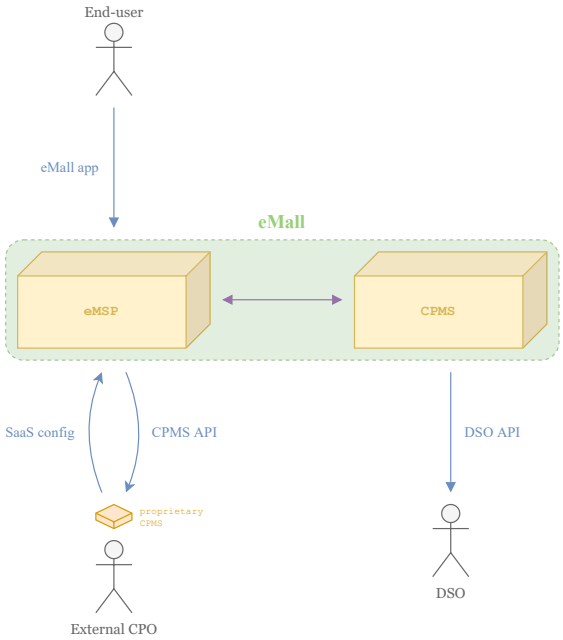

Figure 1.1: A brief and high level description of the eMall project.

- It cannot be excluded that the compliant CPOs own a proprietary eMSP[1] (e.g., deployed as a mobile or web app).
- Similarly, the aforementioned CPOs, as part of the industry, should have a communication channel towards the DSOs.

Finally, external CPSMs are assumed to be built with a process that realizes a product conformed to the interface of the eMSP subsystem of eMall and the documentation of the underlying API.

Identifier	Goal description	Subsystem
G_1	Allow the users to obtain relevant information about nearby charging stations.	eMSP
G_2	Allow the users to book and cancel a charge in advance from their device or directly at the charging point.	eMSP
G_3	Allow the users to start and keep track of the charging process.	eMSP
G_4	Allow the users to pay for a booked charge.	eMSP
G_5	Allow the users to receive personalized suggestions about charging points and time frames for charging their vehicle.	eMSP

[1] However, this would neglect the benefits which eMall tries to yield.

G_6	Allow the client CPO to monitor the internal and external status of its charging stations.	CPMS
G_7	Allow the client CPO to manage the charging process at each of its sockets.	CPMS
G_8	Allow the client CPO to obtain information about energy prices by the DSOs.	CPMS
G_9	Allow the client CPO to stipulate energy provision contracts with DSOs.	CPMS
G_{10}	Allow the client CPO to select the energy source to be used at each charging point.	CPMS
G_{11}	Allow the client CPO to launch special offers at any of its charging points.	CPMS
G_{12}	Allow ECPOs to exploit the eMSP functionalities with a SaaS deployment configuration.	eMSP
G_{13}	Allow the client CPO and ECPOs to publish and advertise their charging stations and the corresponding offers.	eMSP/CPMS
G_{14}	Allow the client CPO and ECPOs to take reservations and have users perform charges at their charging stations.	eMSP/CPMS

1.2 Scope

Phenomenon	Controlled by...?	Shared?	Subsystem
Humans own electric vehicles.	World	N	
Vehicles are driven by humans and move around.	World	N	
The battery of an electric vehicle runs out over time.	World	N	
Charging stations exists in urban and sub-urban settings.	World	N	
The battery of an electric vehicle recharges when plugged in a charging station socket.	World	N	
CPOs own charging stations.	World	N	
DSOs sell energy to CPOs.	World	N	

The user asks the system to show relevant data about the charging stations.	World	Y	eMSP
The CPO asks the system to show information about the charging stations.	World	Y	CPMS
The user asks the system to book a charge.	World	Y	eMSP
The user connects their vehicle to a charging point.	World	Y	CPMS
The user asks the system to trigger the charging process.	World	Y	eMSP
The system manages the charging process.	Machine	Y	CPMS
The user asks the system to show data about the charging process.	World	Y	eMSP
The user pays for a charge.	World	Y	eMSP
The CPO asks the system to show information about the DSOs' energy prices.	World	Y	CPMS
The CPO asks the system to buy a DSO's energy.	World	Y	CPMS
The system gets data from DSOs.	Machine	Y	CPMS
The DSOs send data to the system.	World	Y	CPMS
The CPO asks the system to setup the energy source of a socket.	World	Y	CPMS
The ECPO asks the system to configure its proprietary CPMS.	World	Y	eMSP
The system gets data from ECPOs.	Machine	Y	eMSP
The system notifies personalized suggestions to the user.	Machine	Y	eMSP

1.3 Definitions, acronyms, abbreviations

1.3.1 Definitions

Software as a service (SaaS) is a delivery model in which software is not purchased and installed on individual computers or servers, but is rather access in a cloud-based way. SaaS providers host and maintain the software, and users pay a subscription fee to use it.

1.3.2 Acronyms

Acronym	Extended concept	Meaning
e-Mobility	Electric mobility	A sustainable kind of transport with respect to social and environmental impacts, particularly concerning the reduction of the human ecological footprint.
EV	Electric vehicle	A vehicle adopting electric motor(s) for propulsion, either powered by a collector system (i.e., with electricity from extravehicular sources) or autonomously by a battery.
eMall	e-Mobility for all	The application to be engineered and developed.
CS	Charging station	A piece of equipment that furnishes electrical power for charging plug-in EVs. Also known as charge point (CP) or electric vehicle supply equipment (EVSE).
CSS	Charging station socket	A physical device exposing a port with a connector specifically designed to be plugged in an EV. When grouped together with other CSSs, the whole structure forms a CS.
CPO	Charging point operator	A company owning CSs (CPs, or EVSEs) in the e-Mobility industry.
ECPO	External charging point operator	An external CPO compliant with the eMall project consortium.
eMSP	e-Mobility service provider	The dispense of a service implementing the provisioning of EVs charging services for the final customers.

CPMS	Charge point management system	A system tailored to manage a CPO's IT infrastructure. It handles the acquisition of energy from 3rd party DSOs, distributing it to EVs, and supports decision making.
DSO	Distribution system operators	An energy supplier consulted by a CPO's CPMS.
API	Application programming interface	A set of operations provided by a computing system which can be called by other machines.

1.3.3 Abbreviations

- i.e., or *id est*, is a Latin phrase that translates to «that means» in the sense of «that means»;
- e.g., or *exempli gratia*, is a Latin phrase that translates to «for the sake of example».

1.4 Revision history

– v1.0 (December 23, 2022): initial release.

1.5 Reference documents

- RASD project specifications for A.Y. 2022-23;
- M. Jackson, P. Zave, *The World and the Machine*, 17th International Conference on Software Engineering, 1995, pp. 283-283, doi: 10.1145/225014.225041;
- Unified Modeling Language (UML) specification (see official website);
- Alloy 5.1.0 documentation (see official website) and [1].

1.6 Document structure

1. *Introduction*: a brief presentation of the context of problem, along with goals and phenomena descriptions and some useful notions cited in this document.
2. *Overall description*: a discussion about the requirements and assumptions of the problem, together with some UML diagrams (e.g., class diagram, state diagrams).

3. *Specific requirements*: a detailed overview of Chapter 2 through use cases and related diagrams.
4. *Formal analysis using Alloy*: a formalization of the problem and the software with a model checking tool.

2 | Overall description

2.1 Product perspective

2.1.1 Scenarios

1. **Unregistered EV user signs up to eMall**
   Luca has an electric car, which he uses to commute to work every day. He decides to sign up for eMall in order to reduce his charging expense. He launches the application and inputs his personal information along with a username and password; after validating the input, eMall responds with a success message and Luca can now log in.

2. **eMall user evaluates charging options,then books a charging in a future time frame and performs it**
   Alessandra has a meeting across town in the afternoon, and she realizes her EV will not have enough battery to go back home at the end of the day. Before driving to the location of the meeting, she opens the eMall app on her smartphone and searches for charging points in that area on the map. The application displays two charging points within 1 km of the meeting location: one is operated by CPO A and is located in the parking lot right next to the building where Alessandra's meeting is held, whereas the other, operated by CPO B, is a few blocks away, but it is cheaper thanks to a special offer and it has immediate availability.
   After browsing through the nearby charging points, Alessandra books a charging at the second charging point, making sure to reserve a spot until after her meeting is over. She drives to the CP, plugs the charging cable into her car and starts the charge using the eMall app after entering her credit card payment details. Soon after Alessandra's meeting has finished, she receives a notification of charge completion from eMall: she walks to the charging point, unplugs her EV and drives home.

3. **eMall user books and performs a charging on the fly**
   Matteo is driving to a dentist appointment for a routine checkup, and while he is

looking for a parking spot he notices a charging point close to the dentist's office. He parks in front of a fast socket and selects the "Charge with eMall" option on the socket's touch screen. The screen displays a QR code, which Marco scans on the eMall mobile app: he reserves his spot for the next hour, enters his payment details, and starts the charge after plugging in his EV. He goes to his appointment, exits, then waits until eMall sends a notification of charge completion, after which he unplugs his car and drives home.

4. **CPO manager choosing a convenient DSO**
   Laura, the Lombardia regional manager for CPO A, has to manage energy provisioning for the next weeks. In order to do so, she accesses A's CPMS through the web application and retrieves data about the price and availability of energy from a slate of DSOs $D_1, ..., D_n$. She sorts the results based on the lowest price and does not exclude DSOs which offer contracts with a low total energy volume, since she wants to purchase energy from a mix of different DSOs. Based on the results, Laura elects to purchase 1.8 GWh of energy from D_3 and 1.2 GWh from D_7.

5. **CPO manager deciding the energy mix for CPs**
   Andrea is the manager of all charging points in Milan for CPO C, and he has to choose the energy mix used by each of C's charging points as part of his weekly planning. Andrea accesses C's CPMS and instructs all charging points to use 150 MWh of the energy provided by the DSO supply, then turn to their batteries until they are at 50% and eventually go back to automatic management.

6. **eMall sending a recommendation notification to a user**
   Mattia is an eMall user. Today he has two meetings in different buildings, with a lunch break in between: eMall notices the break based on the events in his calendar, and also finds out about a special offer going on in a charging point which happens to be along the way between the two meetings. As a consequence, when Mattia goes out for the first meeting he receives a notification from eMall advising him to book a charge during his break to save time and still be able to make both meetings while saving on his charge.

2.1.2 Class diagram

There are two main types of actors (`Person`) interacting with eMall, the `User` class refers to EV users while `Manager` refers to the CPO managers who have direct access to the CPMS and to its functionalities.

In this diagram the user is allowed to have more than one vehicle but, as a simplification,

each eMall account can only be linked to a single vehicle during the registration (see *EV user registration*).

The `EnergyMix` class represents a way to describe the pipeline for when the CPO manager selects manual settings for the energy mix (see *Selecting energy mix for a CP*).

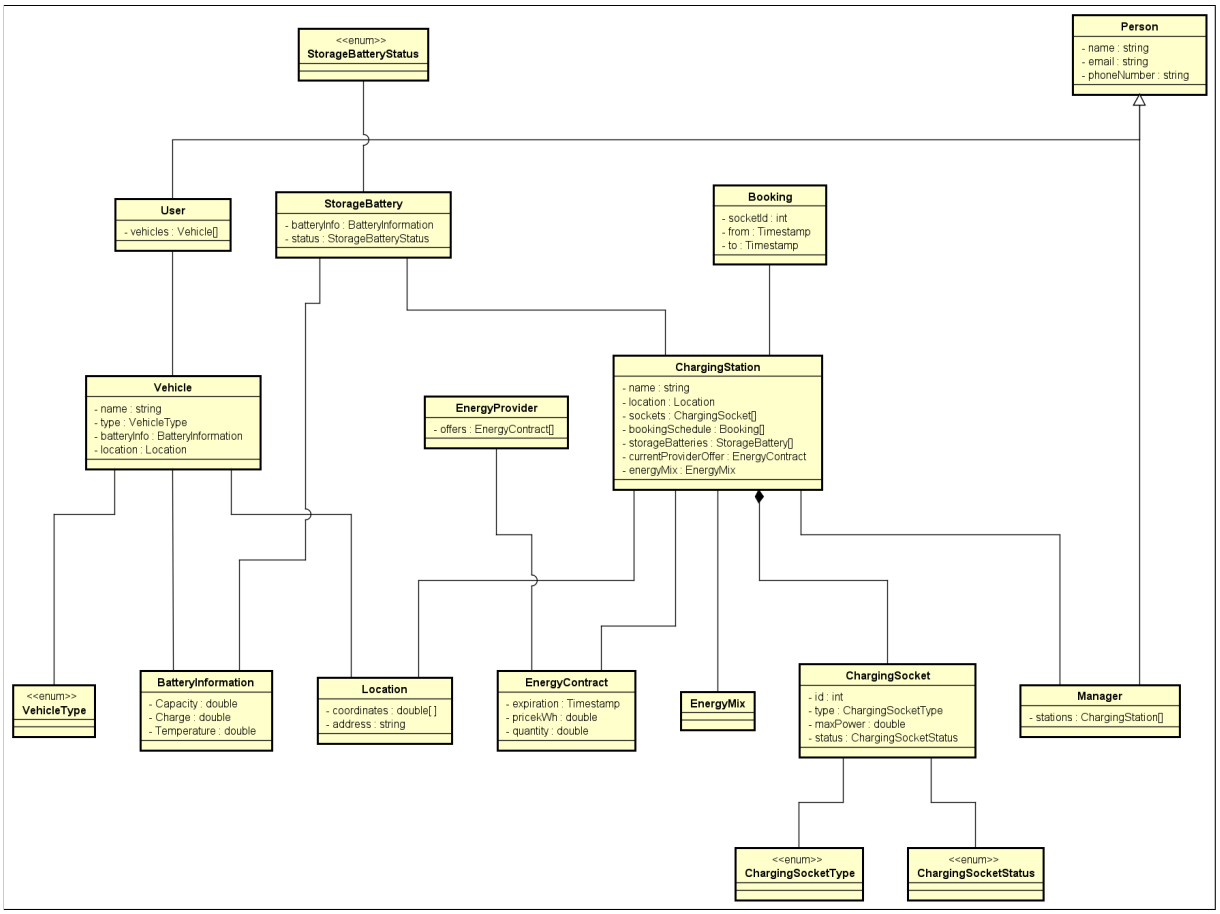

Figure 2.1: A high level class diagram.

2.1.3 Statemachine diagrams

- Figure 2.2 shows how the end user can receive personalized suggestions on their homepage, get information from CSs and book timeframes for a charge[2], and recharge their EV after the setup of a payment method, apart from navigating the app (`change view` trigger).
- Figure 2.3 explains the communication from eMall's eMSP to ECPOs through the external CPMS API (see Figure 1.1).

[2]The two states cooperate with each other.

- Figure 2.4 shows how the client CPO can monitor its CSs and administer their charging processes, together with energy sources, and purchases from DSOs in either manual or automatic way.
- Figure 2.5 clarifies Figure 2.3 contrariwise, the communication from the ECPOs to eMall's eMSP (see Figure 1.1). The former resorts to a subscription mechanism to register its CPMS in the app, enabling the latter's service on its IT infrastructure.

2.2 Product functions

eMall should be able to facilitate several actions carried out by different stakeholders. The most important and critical ones are listed in the paragraphs below after a short summary.

2.2.1 End users

End users must be offered the possibility to sign up for and log into the application, which will be the entry point to book and trigger EV charges. These features are to be mainly accomplished by the eMSP subsystem.

Sign up and login

The sign up and login functions should be implemented like the solutions proposed in usual software engineering contexts. This can be achieved by showing an appropriate form on the UI and storing the e-mail and password pairs in a proper system (e.g., a database).

A user will also need to insert information about their owned EV. For the sake of simplicity, every account is allowed to associate a single vehicle during the setup process.

Booking a charge

Reserving a CSS can be done in two moments, either by effectively booking a spot in the future or on the fly. The minimum bookable temporal slice is 30 minutes, and a reservation can only start at specific time intervals: the top of the hour or half past the hour. Users can book multiple consecutive slices up to a maximum of four, corresponding to 2 hours in total. This choice is justified by two reasons: the CPO needs to widen its consumer base as much as possible and users should be encouranged to let other interested people enjoy the charge feature.

The immediate booking option is devised for consumers who are willing to use unoccupied sockets on the spot, and prompts the user to scan a QR code displayed by the CSS itself to confirm their identity and intention. This flow of events is necessary to let eMall manage this special case coherently with respect to the common and aforementioned one.

A user is also allowed to manually cancel a reservation at any time before triggering the start of the charge.

Start and end of a charge

The charge of an EV must be triggered by the user who performed the booking within 15 minutes following the start of the time frame. Otherwise, the system will automatically cancel the booking and mark the socket as available for the next temporal slice.

Each charge stops when the EV's battery is full or the reserved time frame has ended, whichever occurs first. In either case, the customer receives an end-of-charge push notification from the mobile app. If the charge fully completes before the allotted time expires, then the user can decide to unplug their EV after receiving the message or wait until the end of the time frame.

Paying for a charge

Prior to the start of a charge, the customer is asked to input a valid payment method or select one which they used previously. At the end of the charge, the transaction is verified and processed by eMall with the help of a third party payment gateway (e.g., Braintree, Stripe or Square, all of which allow users to pay with credit/debit cards and virtual wallet services such as Apple Pay or Google Wallet).

Users are allowed to save a payment method in order to reuse it in the future.

Receiving personalized suggestions

eMall's eMSP subsystem can proactively suggest a customer to perform a booking based on their calendar and GPS position.

For example, when a special offer is introduced by a CPO and the users's agenda matches the offer period with a suitable empty slot, the app can generate a push notification to warn the customer. A recommendation can also be emitted when the user is geographically close to one of their most used CSs or the last charge of their EV has happened a long time ago.

2.2.2 Client CPO and DSOs

DSOs provide their energy supplies and make them available to CPOs. The CPMS subsystem has to interact with a standardized API to fetch and process data about prices or any other useful information to guide a proper decision-making mechanism.

Feature	Manual	Automatic
Web app access	Y	N
Monitoring the status of charging stations	Y	N
Stipulating energy contracts with DSOs	Y	Y
Distributing energy to stations	N	Y
Managing charging at sockets	N	Y
Selecting energy source mix	Y	Y
Setting charge costs	N	Y
Setting special offers	Y	Y

Table 2.1: A summary of Section 2.2.2's functions possible modes.

Web app access

Given that the client CPO is not to be considered as an external company and will actually own the eMall system as a whole, it should be noted that no sign up functionality for the CPO will be taken into consideration.

The insertion and deletion of new organization members into the system would be a responsibility of the database administrator or an equipollent position. This can also enable the company to take advantage of a role-based access control (i.e., an approach to restricting the access to authorized personnel) implemented by most of the commercial database management systems.

Monitoring the status of charging stations

Human operators and the system itself must be able to obtain information about the external and internal status of the CSs.

The external status of a CS includes:

- its location (i.e., latitude and longitude);
- the number of sockets and their associated characteristics (e.g., the type of connector or whether it is occupied);

- any other useful information (e.g., the estimated time until a socket will be freed).

Instead, the internal status of a CS includes:

- the level of its batteries, if they are present;
- the number of connected EVs;
- any other useful information (e.g., for each EV experiencing a charge, the energy consumption and the estimated time to reach the end of the process).

Stipulating energy contracts with DSOs

The client CPO can stipulate contracts for energy provision with various DSOs. This task can be handled automatically by eMall's CPMS, choosing the most convenient DSO based on a customizable metric. However, a human operator can take over to manually pick the best DSOs. It is assumed that multiple contracts with different DSOs can be active at any time.

Distributing energy to stations

The energy which can be employed by the client CPO's CSs must be collected in a central storage unit or similar structure owned by the CPO themselves. Furthermore, a power line system should be properly installed to let eMall's CPMS handle the distribution of the energy from the stock to the stations.

Managing charging at sockets

The energy supplied by each socket owned by the client CPO during the charging process is automatically managed by the CPMS. The socket uses the energy source specified by the *Selecting energy source mix* feature and supplies the maximum power that can be transferred by the specific connector type.

Selecting energy source mix

To power the charging process of an EV, a CS can rely on its batteries or on a DSO's energy bought by the client CPO. As a consequence, the CPMS should let human operators decide a proper mix of sources for the station.

Among the possibilities, this can be achieved with the definition of a workflow involving a sequence of actions such as using a certain amount of a source or for a certain given of time. This implementation could be enhanced with conditionals (i.e., if-then statements)

and loops (e.g. using a source until a the batteries reach a limit percentage) to define an high level programming language. Any exception or inability to accomplish the specified steps would turn the system to an automated behaviour.

Setting charge costs

Managing a big set of CSs is overwhelming, hence eMall's CPMS will dynamically infer a suitable cost for a charge depending on parameters like the current DSOs' energy prices, the adopted energy mix and other operational costs.

Setting special offers

Limited time offers are a strategic method to tempt the users to charge their EVs at a convenient price. An employee of the client CPO can set a discount rate for one or more of their CSs, and the CPMS can automatically launch smart offers based on current energy price, user behavior statistics and other available data.

2.2.3 ECPOs

External CPOs can advertise their CSs and the related CSSs with the help of the communication between their CPMS and the eMSP subsystem. The former has to exhibit a standardized API, while the latter a management dashboard.

Sign up

The external companies interested in the project will need to request a consult with eMall's corporate relations team[3].

In collaboration with eMall's accounting team, a SaaS subscription plan will be designed and proposed to the ECPO. This can result, for example, in a monthly or annual membership to be renewed in the future. A subscription is necessary to legitimately access and communicate with eMall's eMSP service.

On the other hand, in order to let eMall establish a proper connection towards the external CPMS(s), the ECPO will be asked to provide their API endpoint and authentication keys during the consult.

After the ECPO signs up for a subscription by accepting a plan, eMall's team will generate

[3]Unlike *Sign up and login* and *Web app access*, this other setting's perspective can be circumscribed in a business-to-business (B2B) model. Notice that, in particular, *Sign up and login* refers to a business-to-customer (B2C) model.

the credentials, which will be the one and only entry point to the SaaS system for the whole ECPO organization[4].

SaaS configuration

A registered and actively subscribed ECPO is eligible to access eMall's SaaS dashboard. The login brings the ECPO to a landing page where two main operations can be carried out: the configuration of a new CPMS and parameter tuning.

The first option lets eMall connect to an external CPMS endpoint and fully exploit a bidirectional communication channel. The second option, instead, allows the ECPO to update and/or delete a present configuration (e.g., revise an API key).

2.3 User characteristics

- **Unregistered EV user**
  A person who uses an EV but is not registered to eMall yet, hence they have to sign up to the service in order to access its functionalities.
- **eMall user**
  An EV user who is registered to eMall and uses the mobile app to access features such as booking and paying for a charge and receiving personalized suggestions about when and where to charge their EV.
- **CPO manager**
  An employee of the client CPO who is tasked with the management of charging points or energy provisioning. They access the CPMS functionalities through a web app.
- **ECPO employee**
  An employee of an ECPO who is tasked with setting up and configuring the SaaS deployment for eMall's eMSP.

2.4 Assumptions, dependencies and constraints

2.4.1 Domain assumptions

[4]This does not exclude that the ECPO could create "aliases" or route multiple proprietary access methods to the entry point itself. However, this is not part of the scope of the analysis, as it involves the ECPO's internal business decisions.

Identifier	Description
D_1	eMall users have a mobile device with internet access, GPS location and a camera where they can install the application.
D_2	eMall users enter truthful data about their car manufacturer and model.
D_3	eMall users disconnect their EV from the socket within 5 minutes of the expiration of the reserved time slot.
D_4	eMall's eMSP service can fetch information about each user's EV, such as the battery level, through brand-specific RESTful APIs.
D_5	eMall's eMSP service can rely on a third party payment manager to handle all charge payment transactions.
D_6	eMall's eMSP service can communicate with its proprietary CPMS and external CPMSs through RESTful APIs.
D_7	CPOs have a secure and reliable two-way communication channel with their sockets for exchanging messages such as socket status updates and charge start prompts.
D_8	Managers of the client CPO using eMall's proprietary CPMS have a device with a modern browser where they can access the web application.
D_9	Managers of ECPOs using eMall's eMSP service have a device with a modern web browser where they can access the eMall SaaS dashboard.
D_{10}	eMall's proprietary CPMS can fetch information about DSO energy prices and sources through RESTful APIs.
D_{11}	eMall's proprietary CPMS can send information about energy provision contract proposals to DSOs through RESTful APIs.
D_{12}	eMall's proprietary CPMS can rely on a third party payment manager to handle all provision contract payment transactions.
D_{13}	The energy provided to each CPO by DSOs as part of provision contracts is distributed to its CPs through an appropriate infrastructure (e.g. using dedicated cables and power lines), which is maintained and monitored by that single CPO or a group of partner CPOs.
D_{14}	Each CPO has a central storage unit which can store energy and distribute it to CPs later as needed.
D_{15}	The surplus energy from DSO provision for each CPO is stored either in batteries located at every CP or in the CPO's central energy storage unit.

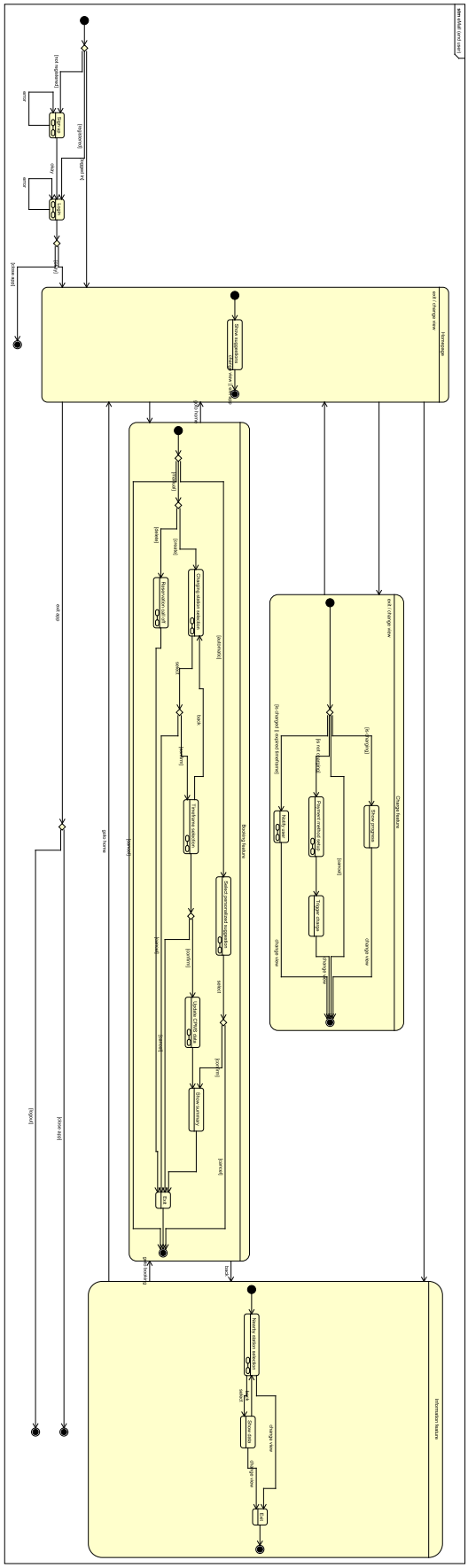

Figure 2.2: The main features of the end user side of eMall's eMSP service.

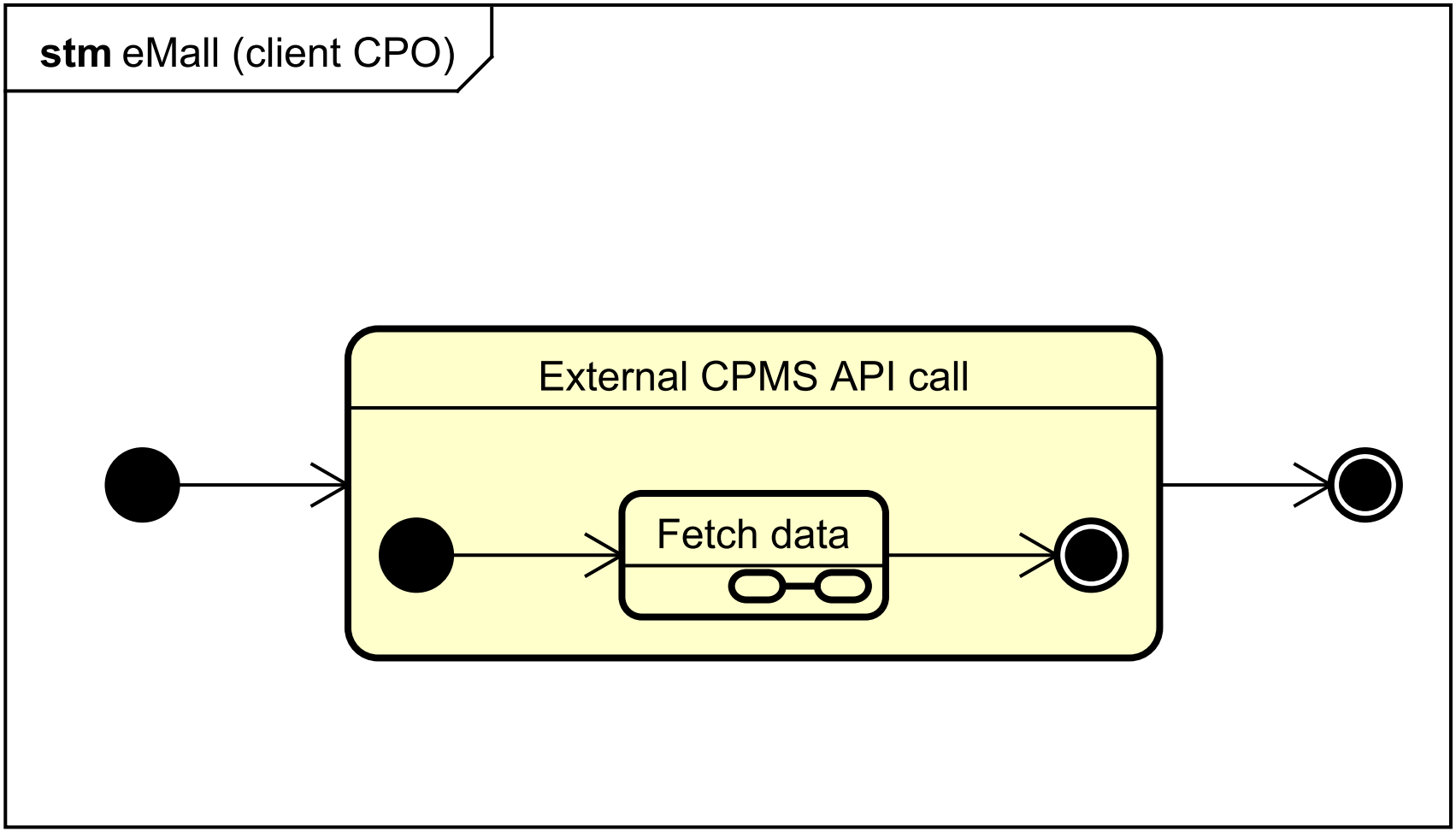

Figure 2.3: A short illustration of the client CPO side of eMall's eMSP service.

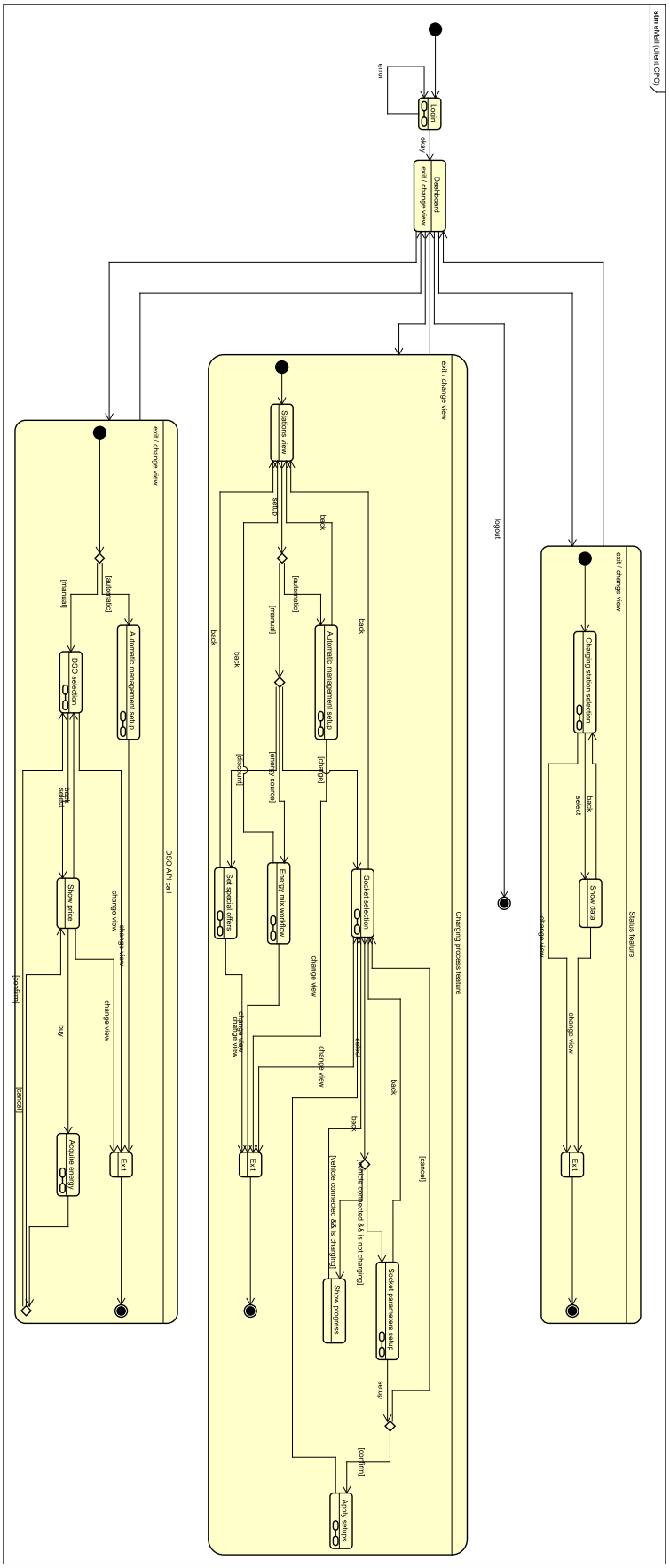

Figure 2.4: The main features of the client CPO side of eMall's CPMS service.

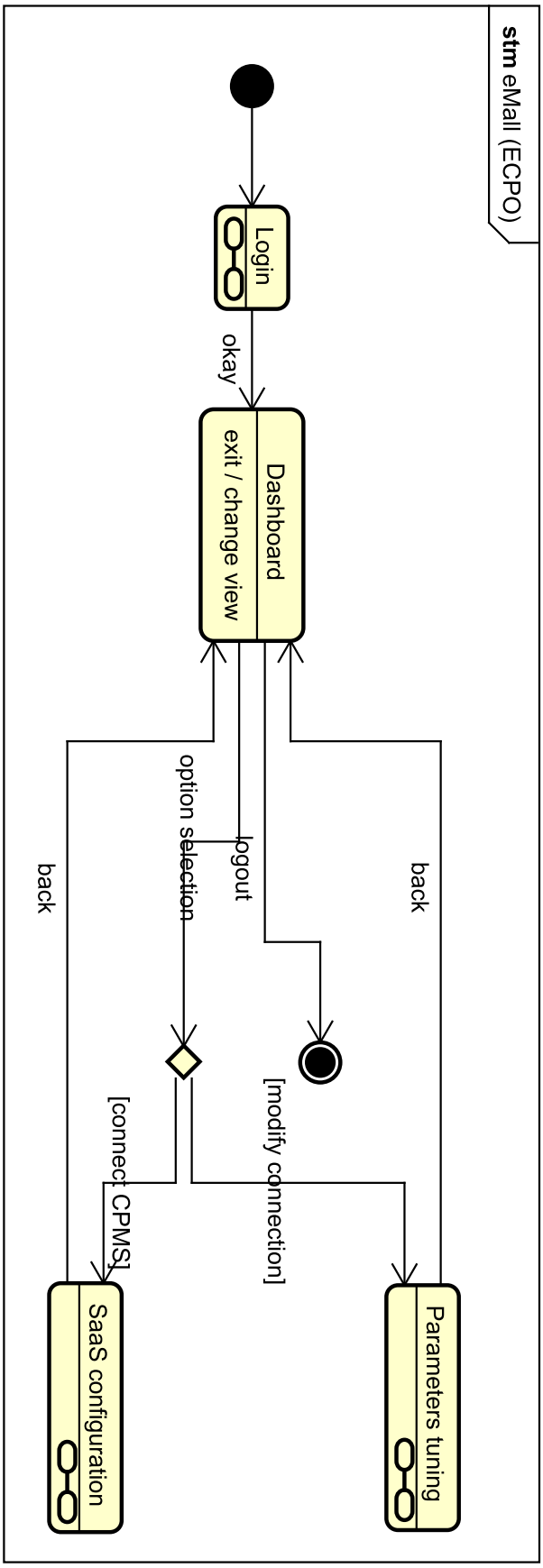

Figure 2.5: A terse depiction of eMall's SaaS feature available to ECPOs.

3 | Specific requirements

3.1 External interface requirements

3.1.1 User interfaces

The mobile app addressed to the end users should be designed to be easy-to-use, including all the main functionalities like a sign up/login page, a landing screen with an overview of the EV status, the reserved charges and appealing special offers.

The client CPO's web app must grant a view on the CSs and ensure that the aforementioned functions in Section 2.2.2 are accessible and/or navigable.

The ECPOs' web app should conceptualize the SaaS model.

3.1.2 Hardware interfaces

CSSs' connectors and EVs' ports are the only physical components that need to adequately work together. Their connection must also be able to let the corresponding CPMS infer the supply methodology, direct current (DC) or alternating current (AC), and suitably inject the required electric power.

3.1.3 Software interfaces

Omitting the APIs that enable the communication between eMall's subsystems and the other business actors, the scan of a QR code is a functionality which describes the interaction between the end user's mobile device and the CSSs, expressly the eMall app and the display of the CSSs.

3.1.4 Communication interfaces

The CPMS and ECPMSs provide a standardized REST API through which the eMSP can retrieve information about the charging stations and the available sockets and also

manage reservations.

Such API should provide methods to:

- get the list of all the CPs in a certain area;
- get the list of charging sockets in a specific CP with their availability and type;
- get the available time slots for a specific charging socket;
- book one or more time slots;
- cancel a reservation;
- get information about the charging process;
- start/stop the charging process;
- retrieve all the notifications from a charging point to all the users.

The latter would be used by the eMSP to periodically (e.g. once per minute) retrieve the notifications from a charging point and send them to the correct users.

The EVs and DSOs are assumed (domain assumptions D_4 and D_{10}) to provide APIs to retrieve information and, in the case of the DSOs, manage energy provision contracts.

3.2 Functional requirements

3.2.1 Requirements

Identifier	Description	Subsystem
R_1	The system shall allow an unregistered EV user to create an account.	eMSP
R_2	The system shall allow a registered EV user to log in.	eMSP
R_3	The system shall allow a registered EV user to browse the available charging points and their sockets.	eMSP
R_4	The system shall allow a registered EV user to reserve a charge for a future time frame at a charging point.	eMSP
R_5	The system shall allow a registered EV user to cancel a reservation for a future time frame.	eMSP
R_6	The system shall allow a registered EV user to reserve a charge on the fly directly at a charging point.	eMSP

R_7	The system shall allow a registered EV user to start a charge during the reserved time frame at the reserved charging point.	eMSP
R_8	The system shall allow a registered EV user to start a charge reserved on the fly at the charging point.	eMSP
R_9	The system shall allow a registered EV user to pay for a charge.	eMSP
R_{10}	The system shall notify a registered EV user when a charge has finished.	eMSP
R_{11}	The system shall notify a registered EV user about charging points with ongoing offers.	eMSP
R_{12}	The system shall notify a registered EV user about free timeframes at charging points that align with the user's agenda.	eMSP
R_{13}	The system shall notify a registered EV user about free timeframes at charging points that align with the user's position and movements.	eMSP
R_{14}	The system shall allow ECPOs to register with valid Saas credentials.	eMSP
R_{15}	The system shall allow ECPOs to configure the CPMS API connection.	eMSP
R_{16}	The system shall allow a manager of the client CPO to log in.	CPMS
R_{17}	The system shall allow the client CPO to retrieve the number, type, cost, and occupation status of a charging point's sockets.	CPMS
R_{18}	The system shall allow the client CPO to retrieve the battery level and charging status of the sockets of a charging point.	CPMS
R_{19}	The system shall allow the client CPO to retrieve the available DSOs' energy prices.	CPMS
R_{20}	The system shall allow a manager of the client CPO to stipulate energy provision contracts with DSOs.	CPMS
R_{21}	The system shall allow a manager of the client CPO to select which DSO under contract should provide energy to a specific charging point.	CPMS

R_{22}	The system shall allow a manager of the client CPO to specify how a charging point should manage the energy mix between battery and DSO provision.	CPMS
R_{23}	The system shall allow a manager of the client CPO to create special offers at one or more charging points.	CPMS
R_{24}	The system shall allow the client CPO to prompt energy emission at a specific socket.	CPMS
R_{25}	The system shall allow the client CPO to monitor the process of charging an EV at a specific socket.	CPMS
R_{26}	The system shall allow the client CPO to automatically stipulate energy provision contracts with DSOs without human intervention.	CPMS
R_{27}	The system shall allow the client CPO to automatically select which DSO under contract should provide energy to a specific charging point without human intervention.	CPMS
R_{28}	The system shall allow the client CPO to automatically specify how a charging point should manage the energy mix between battery and DSO provision without human intervention.	CPMS
R_{29}	The system shall allow the client CPO to automatically create special offers at one or more charging points without human intervention.	CPMS
R_{30}	The system shall communicate the present and future occupation status of the client CPO's sockets when prompted.	CPMS
R_{31}	The system shall communicate ongoing offers at the client CPO's charging points when prompted.	CPMS
R_{32}	The system shall update the occupation status of the client CPO's sockets when a reservation is made.	CPMS
R_{33}	The system shall update the occupation status of the client CPO's sockets when a reservation is canceled.	CPMS
R_{34}	The system shall start a reserved charge at one of the client CPO's sockets when prompted.	CPMS
R_{35}	The system shall communicate the charging status of an EV plugged into one of the client CPO's sockets when prompted.	CPMS

3.2.2 Mapping on goals

Goal	Domain assumptions	Requirements
G_1	D_1, D_6, D_7	R_1, R_2, R_3
G_2	D_1, D_3, D_6, D_7	R_1, R_2, R_4, R_5, R_6
G_3	D_1, D_2, D_3, D_4, D_6, D_7	R_1, R_2, R_7, R_8, R_{10}
G_4	D_1, D_5, D_6, D_7	R_1, R_2, R_9
G_5	D_1, D_2, D_4, D_6, D_7	R_1, R_2, R_{11}, R_{12}, R_{13}
G_6	D_7, D_8	R_{16}, R_{17}, R_{18}, R_{32}, R_{33}
G_7	D_7	R_{16}, R_{17}, R_{18}, R_{24}, R_{25}, R_{34}, R_{35}
G_8	D_8, D_{10}	R_{16}, R_{19}
G_9	D_8, D_{10}, D_{11}, D_{12}, D_{13}, D_{14}, D_{15}	R_{16}, R_{20}, R_{26}
G_{10}	D_7, D_8, D_{13}, D_{14}, D_{15}	R_{16}, R_{21}, R_{22}, R_{27}, R_{28}
G_{11}	D_7, D_8	R_{16}, R_{23}, R_{29}
G_{12}	D_9	R_{16}, R_{17}
G_{13}	D_7	R_{30}, R_{31}
G_{14}	D_7	R_{31}, R_{32}, R_{33}, R_{34}

3.2.3 Use cases

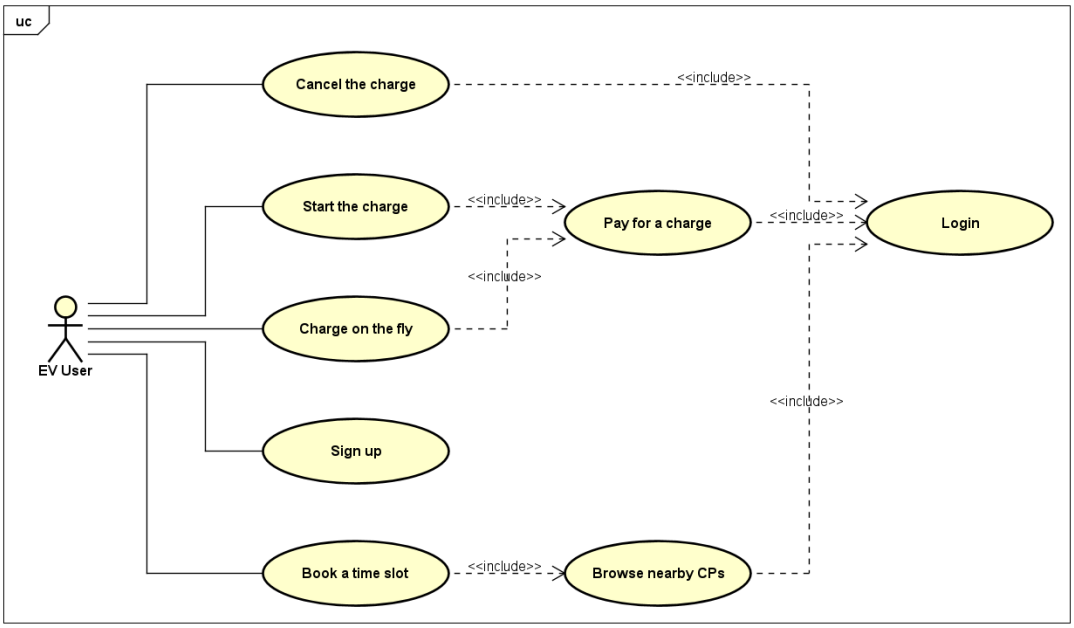

Figure 3.1: The use case diagram of an EV User.

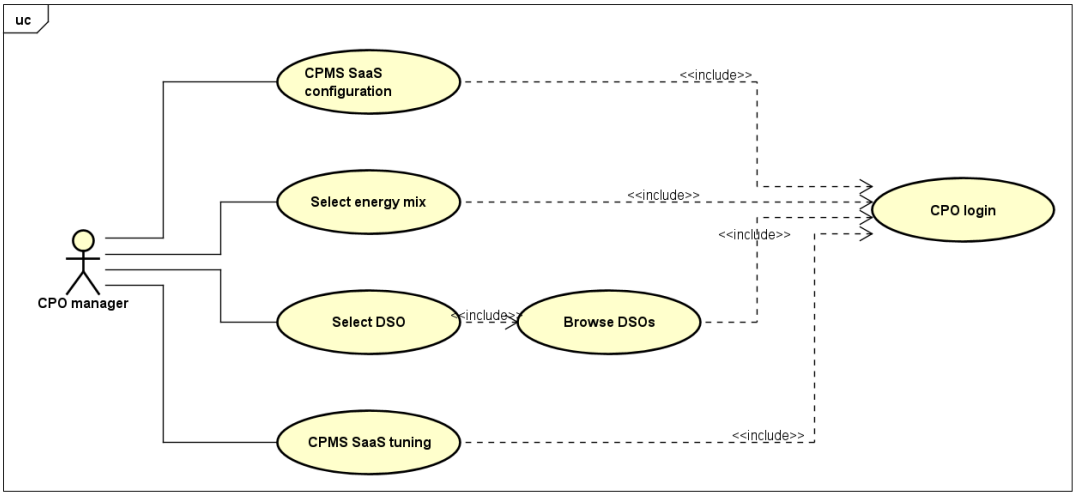

Figure 3.2: The use case diagram of a CPO manager.

EV user registration

Actor: unregistered EV user.

Entry conditions

- The user does not have an eMall account yet.
- The user has downloaded the eMall mobile app and is on the initial registration/login view.

Event flow

1. The user taps the "Sign up" button.
2. eMall displays a form where the user has to input their first name, last name, date of birth, email address, and a password for their account, and then repeat the password to make sure it was typed correctly.
3. The user fills in the form and taps the "Next" button.
4. eMall displays a form where the user has to input the model and license plate number of their EV (if the user's car is not present in the list of models, they can manually input the name and battery capacity).
5. The user fills the form and taps the "Sign up" button.
6. eMall processes the data, creating the new account, and displays the login view with a success message prompting the user to log in.

Exceptions

- *The user fails to input some of the required information:*
  eMall displays the same view, highlighting the missing fields and specifying that they have to be filled.
- *The user inputs invalid information, such as an email address which is not correctly formatted, two different passwords, or an inexistent license plate number:*
  eMall displays the same view, highlighting the invalid fields and specifying why they were not accepted.
- *The user inputs an email address which is already associated with an eMall account:*
  eMall displays the same view, highlighting the email field and specifying that it is already associated with an account.

Activity diagram: Figure 3.3.

eMall user login

Actor: eMall user.

Entry conditions

- The user is registered to eMall.
- The user has downloaded the eMall mobile app and is on the initial registration/login view.

Event flow

1. The user taps the "Sign in" button.
2. eMall displays a form where the user has to input the email address and password associated with their account.
3. The user fills in the form and taps the "Sign in" button.
4. eMall validates the data, creating a user session on the app, and displays the main view.

Exceptions

- *The user fails to input some of the required information:*
  eMall displays the same view, highlighting the missing fields and specifying that they have to be filled.
- *The user inputs an incorrect password:*
  eMall displays the same view, highlighting the password field and specifying that is

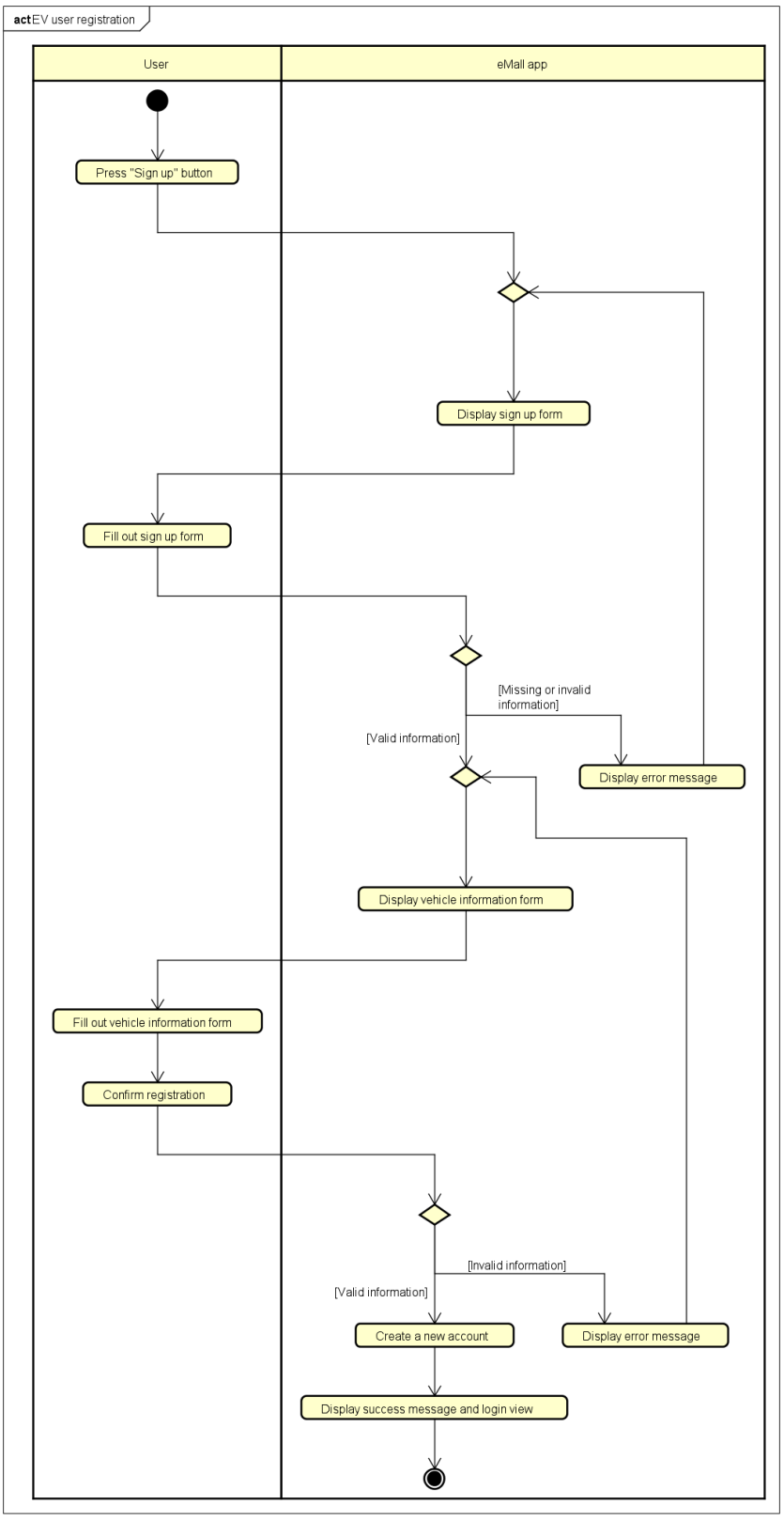

Figure 3.3: EV user registration activity diagram.

is not correct.

- *The user inputs an email address which is not associated with an eMall account:* eMall displays the same view, highlighting the email field and specifying that the email is not associated with any account.

Activity diagram: Figure 3.4.

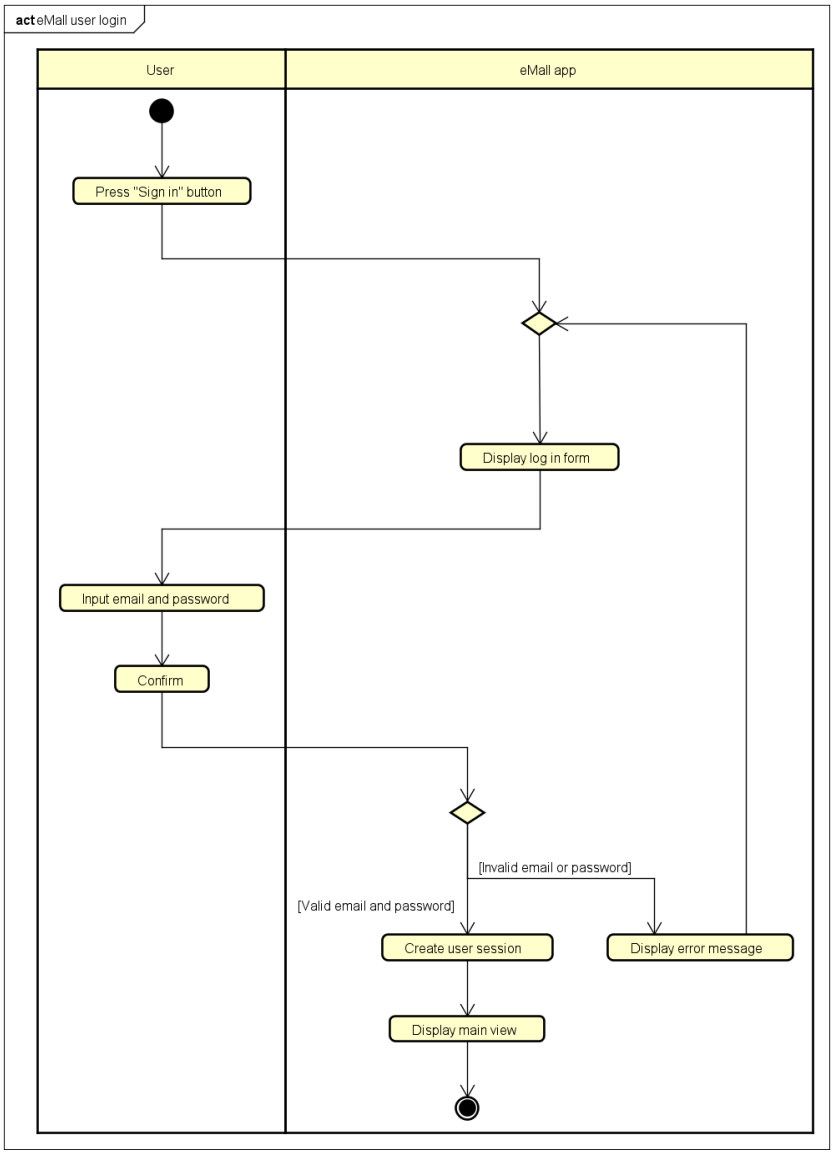

Figure 3.4: eMall user login activity diagram.

Browsing nearby CPs and the relative charging options

Actor: eMall user.

Entry conditions

- The user is registered to eMall.
- The user has downloaded the eMall mobile app and is on the initial home view.

Event flow

1. The user taps the "Nearby CPs" button.
2. eMall diplays a map view, which highlights the user's location and the closest charging points; the marked charging points update as the map is moved, and initially the results are automatically filtered to only show CPs equipped with sockets matching the user's connector type.
3. The user taps on the "Filter" button to change the search parameters.
4. eMall displays an overlay with the available filtering options.
5. The user selects the desired price range and taps the "Confirm" button.
6. eMall hides the overlay, then updates the search results to match the filters and shows them on the map.

Exceptions

- *The user has not authorized eMall to access the smartphone's location:*
  eMall displays an alert prompting the user to grant the required permissions for the app to function properly, then returns to the map view if such permissions have been granted.

Activity diagram: Figure 3.5.

Booking a time slot at a CP

Actor: eMall user.

Entry conditions

- The user is registered to eMall.
- The user has downloaded and logged into the eMall mobile app, and is on the nearby CPs view.

Event flow

1. The user taps on the preferred CP.
2. eMall displays a CP detail view with the CP's address, its distance from the current location, an alert explaining currently active offers, and a list of the available

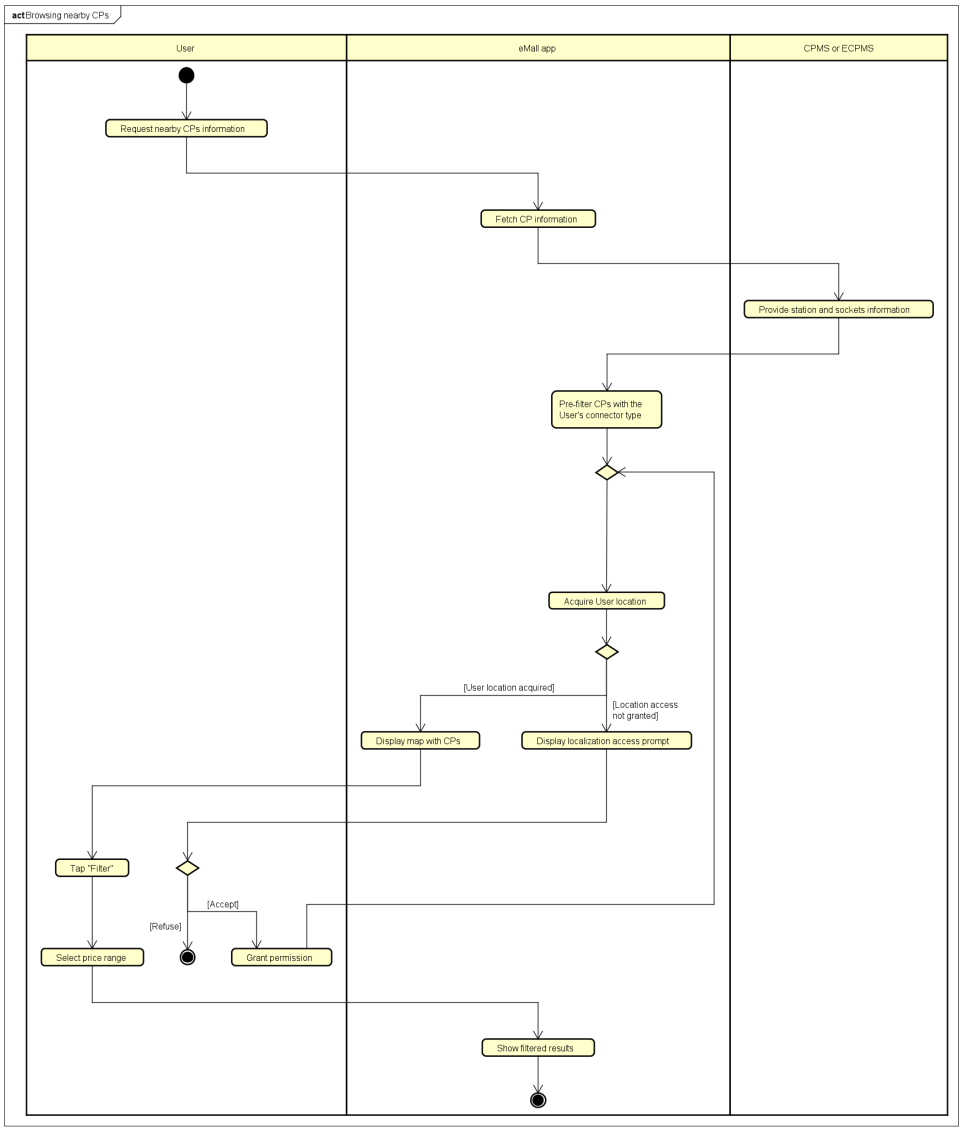

Figure 3.5: Browsing nearby CPs activity diagram.

charging sockets, grouped by connector type and described by the charging speed (slow, fast, or rapid) and charging price (in €/kWh).

3. The user taps on the desired socket type.
4. eMall displays a view containing all available slots for the current day.
5. The user selects a time slot for the charge and taps the "Confirm" button. Each single slot is 30 minutes long, and the user can reserve consecutive slots.
6. eMall processes the booking and displays a view containing details such as CP address, time slot and type of socket reserved.

Exceptions

- *The selected time slot is not available anymore by the time the user confirms the reservation*:
  eMall displays the updated time slot selection view, along with an alert explaining that the time slot is no longer available.
- *eMall cannot establish a connection to the CPMS of the CPO which operates the selected CP*:
  eMall displays an alert signaling the connection error.
- *The CPO's CPMS cannot correctly process the reservation request*:
  eMall displays an alert signaling the CPMS error.

Activity diagram: Figure 3.6.

Booking and starting a charge on the fly

Actor: eMall user.

Entry conditions

- The user is registered to eMall.
- The user has downloaded and logged into the eMall mobile app.
- The user is at an available socket at a CP.

Event flow

1. The user taps on the "Charge with eMall" button on the socket UI.
2. The socket UI displays a QR code needed to sync with eMall.
3. The user opens the eMall app and taps on the "Scan to charge" button.
4. eMall displays a form where the user has to select when they want the reserved time frame to end.
5. The user inputs the desired end time and taps on the "Confirm" button.
6. eMall displays a payment details view, where the user has to insert payment details or select a previously used payment method to pay for the charge.
7. The user inputs all the required information and taps the "Confirm" button.
8. eMall validates the payment information and displays an alert prompting the user to plug their car into the socket.

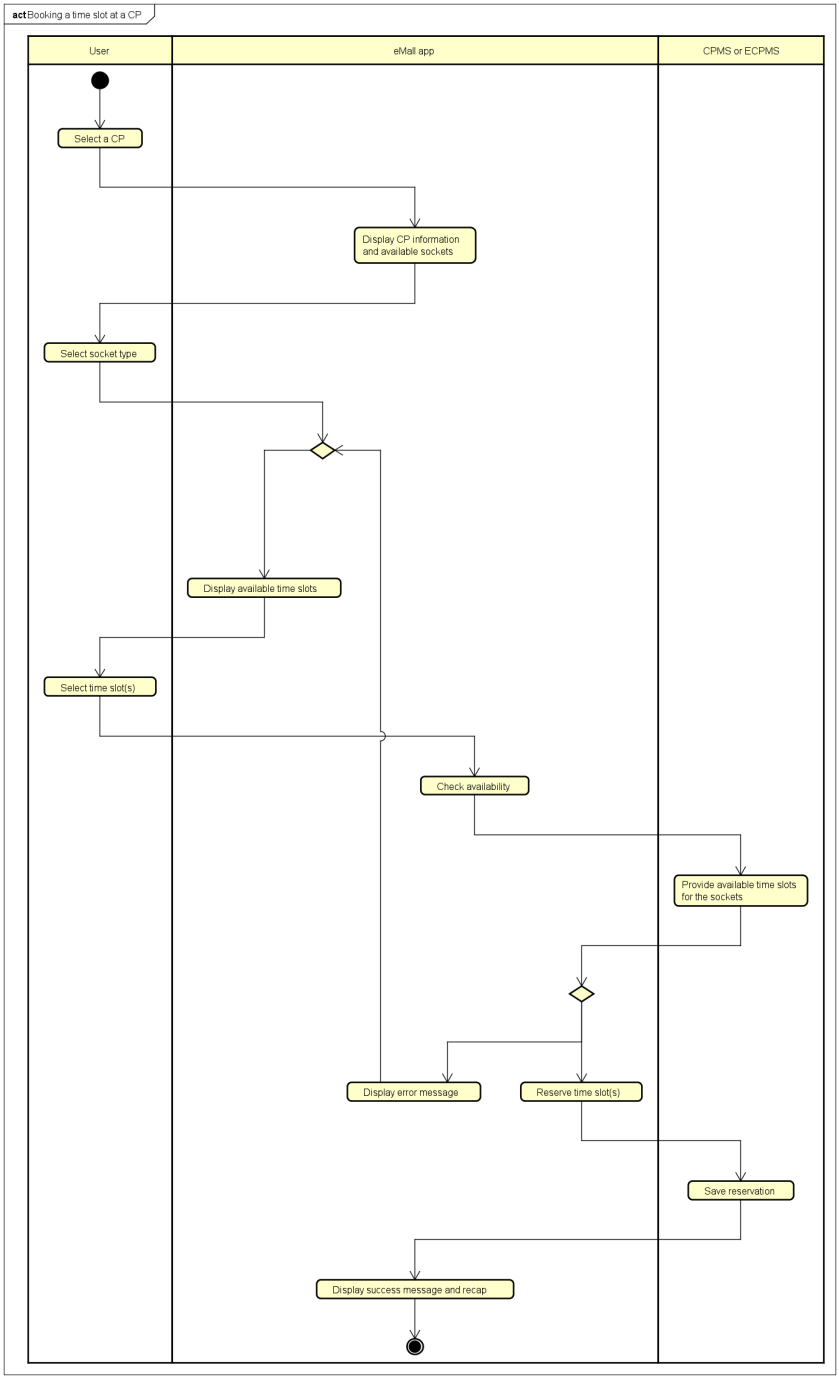

Figure 3.6: Booking a time slot at a CP activity diagram. Note: information about the CP and the available sockets is considered to have already been retrieved by the eMall app during the process of displaying all nearby CPs.

9. The user connects the car to the socket, then taps the "Start" button.
10. eMall communicates with the CPMS of the CP's operator and starts the charge, then displays the reservation details view with a success message confirming that

the EV has started charging.

Exceptions

- *The user selects a time frame which overlaps with a reserved time frame*:
  eMall displays the time frame selection form with an error alert signaling the overlap.
- *The user enters an invalid payment method*:
  eMall displays an error alert asking the user to enter a valid payment method.
- *The user does not correcly connect their car to the socket before starting the charge*:
  eMall displays an error alert prompting the user to reconnect their car and try to start the charge again.
- *eMall cannot establish a connection to the CPMS of the CPO which operates the selected CP*:
  eMall displays an alert signaling the connection error.

Activity diagram: Figure 3.7.

Canceling a reserved charge

Actor: eMall user.

Entry conditions

- The user is registered to eMall.
- The user has downloaded and logged into the eMall mobile app, and is on the home view after having booked a time slot at a CP.

Event flow

1. The user taps on the "Manage" button under the reservation details.
2. eMall displays the reservation details view, which includes the CP name and address, the number, charge speed and connector type of the reserved socket, the "Start charge" button and the "Cancel reservation" button.
3. The user taps the "Cancel reservation" button.
4. eMall displays a confirmation alert, asking the user whether they are sure they want to cancel.
5. The user taps the "Confirm" button.

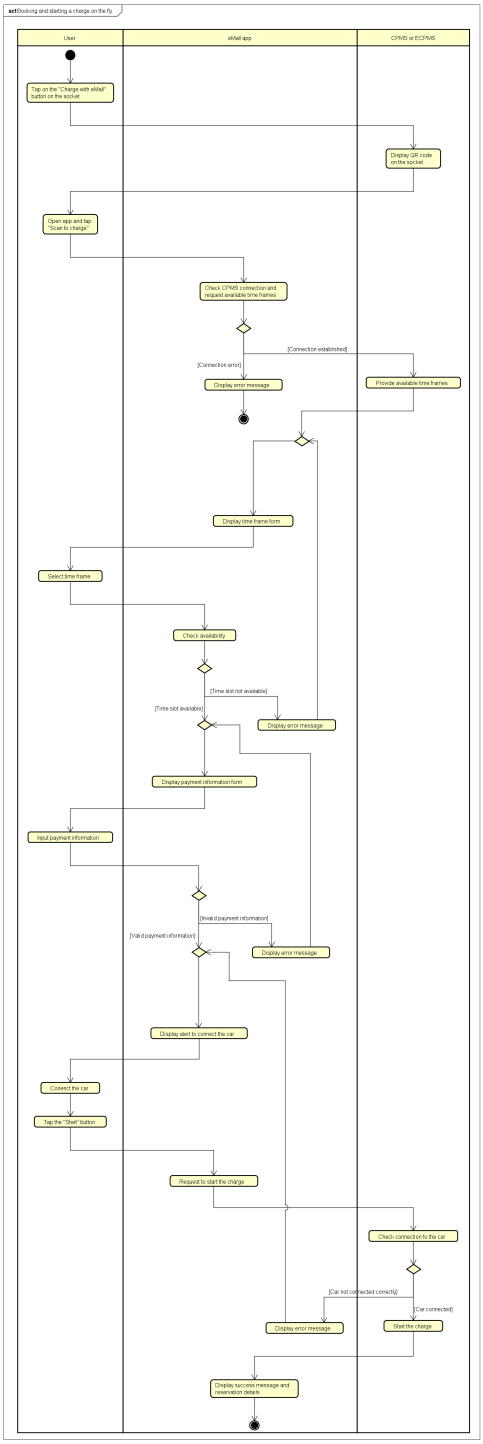

Figure 3.7: Booking and starting a charge on the fly activity diagram.

6. eMall communicates with the CPMS of the CP's operator and cancels the reservation, then displays a message confirming that the operation was successful.

Exceptions

- *The user tries to cancel the charge more than 15 minutes after the start of the allocated time slot*:
  eMall displays an alert explaining that the reservation was already automatically canceled.
- *The user tries to cancel the charge after having already started it*:
  eMall displays an error alert warning the user that the charge has already started and it cannot be canceled anymore.
- *eMall cannot establish a connection to the CPMS of the CPO which operates the selected CP*:
  eMall displays an alert signaling the connection error.

Activity diagram: Figure 3.8.

Starting a reserved charge

Actor: eMall user.

Entry conditions

- The user is registered to eMall.
- The user has downloaded and logged into the eMall mobile app, and is on the home view after having booked a time slot at a CP.

Event flow

1. The user taps on the "Manage" button under the reservation details.
2. eMall displays the reservation details view, which includes the CP name and address, the number, charge speed and connector type of the reserved socket, the "Start charge" button and the "Cancel reservation" button.
3. The user taps the "Start charge" button.
4. eMall displays a payment details view, where the user has to insert payment details or select a previously used payment method to pay for the charge.
5. The user inputs all the required information and taps the "Confirm" button.
6. eMall validates the payment information and displays an alert prompting the user to plug their car into the socket.
7. The user connects the car to the socket, then taps the "Start" button.

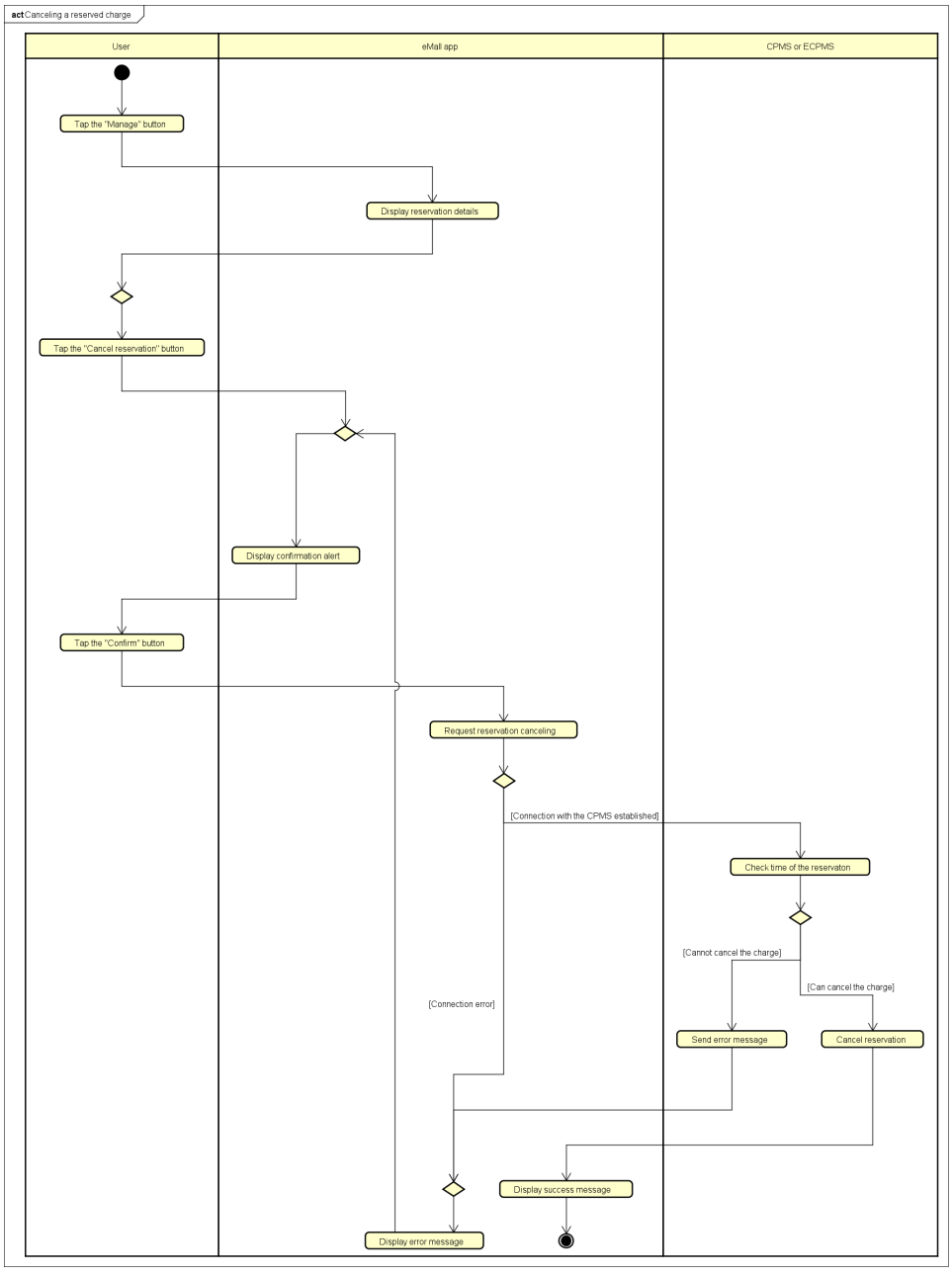

Figure 3.8: Canceling a reserved charge activity diagram.

8. eMall communicates with the CPMS of the CP's operator and starts the charge, then displays the reservation details view with a success message confirming that the EV has started charging.

Exceptions

- *The user enters an invalid payment method*:
  eMall displays an error alert asking the user to enter a valid payment method.

- *The user tries to start the charge before the start of the allocated time slot*:
  eMall displays an error alert explaining that the user has to wait until the time slot starts.
- *The user tries to start the charge more than 15 minutes after the start of the allocated time slot*:
  eMall displays an error alert explaining that too much time has elapsed and the reservation was automatically canceled.
- *The user does not correcly connect their car to the socket before starting the charge*:
  Mall displays an error alert prompting the user to reconnect their car and try to start the charge again.
- *eMall cannot establish a connection to the CPMS of the CPO which operates the selected CP*:
  eMall displays an alert signaling the connection error.

Receiving the end-of-charge notification

Actor: eMall user.

Entry conditions

- The user is registered to eMall.
- The user has downloaded and logged into the eMall mobile app, and has started a charge they booked through the app.
- The time slot reserved by the user has ended, or their EV has charged completely.

Event flow

1. eMall communicates with the CPMS of the CP's operator and interrupts the charge, processes the payment based on the amount of energy used, then sends an end of charge notification to the user.
2. The user taps on the notification, opening the eMall app.
3. eMall displays the reservation details view, which includes the CP name and address, the number, charge speed and connector type of the reserved socket, the total energy emission and the "Close" button.
4. The user taps the "Close" button.
5. eMall displays the home view.

Exceptions

- *The payment does not go through*:
  eMall displays an alert warning the user about the error and prompting them to enter another payment method.

Activity diagram: Figure 3.9.

Receiving personalized suggestions

Actor: eMall user.

Entry conditions

- The user is registered to eMall.
- The user has downloaded and logged into the eMall mobile app.
- The user has given eMall permission to access their calendar and location.

Event flow

1. eMall dynamically computes a slate of suggestions for potential charging points and time slots based on the user's agenda, their movements (using both real-time and historic data), and special offers active at charging points, then sends a notification relaying these suggestions.
2. The user taps on the notification.
3. eMall opens the suggestions view, which includes a list of the suggested CPs and time frames along with the reason why they were suggested.
4. The user taps on the preferred CP, then books a time slot as described above.

Exceptions

- *The user taps on the notification but the suggestion is no longer valid (e.g. the specific offer is no longer available)*:
  Mall displays an error alert signaling that the suggestion has expired.

CPMS SaaS configuration

Actor: ECPO manager.

Entry conditions

- The manager is registered and logged into eMall.

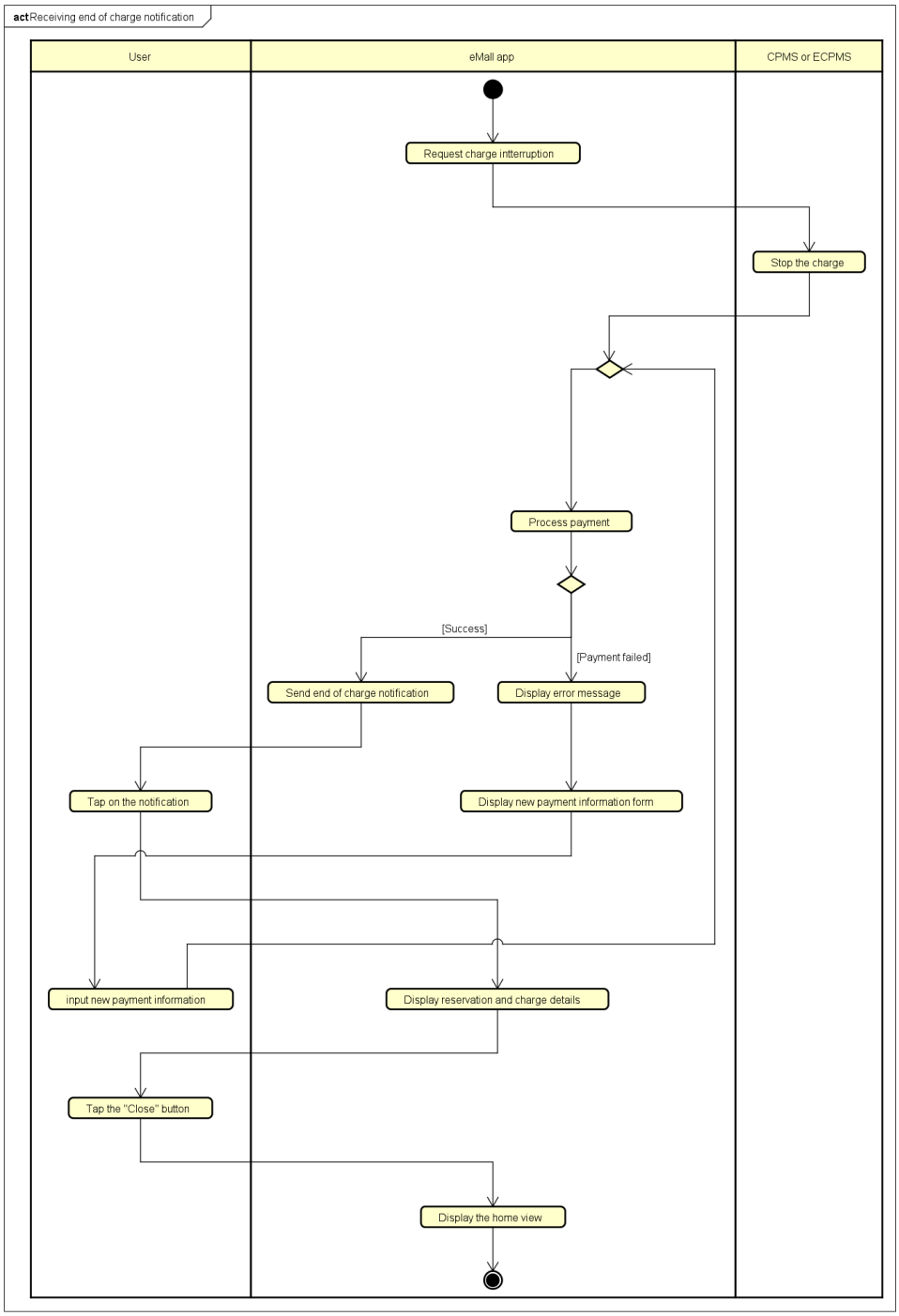

Figure 3.9: Receiving end of charge notification activity diagram.

- The manager is on the the eMall web app dashboard.

Event flow

1. The manager clicks the “Configure” button.

2. eMall displays a form where the manager can enter information about the API used to connect the ECPO's CPMS with eMall (e.g., the API endpoint or the access key).
3. The manager inputs the required information and clicks the "Save" button.
4. eMall displays the web app dashboard with the updated information.

Exceptions

- *The manager inputs an unreachable API endpoint*:
  eMall displays an error alert signaling the inability to establish a connection.
- *The manager inputs an invalid access key*:
  eMall displays an error alert prompting the manager to enter a valid key.

Activity diagram: Figure 3.10.

CPMS SaaS tuning

Actor: ECPO manager.

Entry conditions

- The manager is registered and logged into eMall.
- The manager is on the the eMall web app dashboard.

Event flow

1. The manager clicks the "Modify" button next to the configured CPMS.
2. eMall displays a form where the manager can update or delete information about the API used to connect the ECPO's CPMS with eMall (e.g., the API endpoint or the access key).
3. The manager inputs the updated information and clicks the "Save" button.
4. eMall displays the web app dashboard with the updated information.

Exceptions

- *The manager inputs an unreachable API endpoint*:
  eMall displays an error alert signaling the inability to establish a connection.
- *The manager inputs an invalid access key*:
  eMall displays an error alert prompting the manager to enter a valid key.

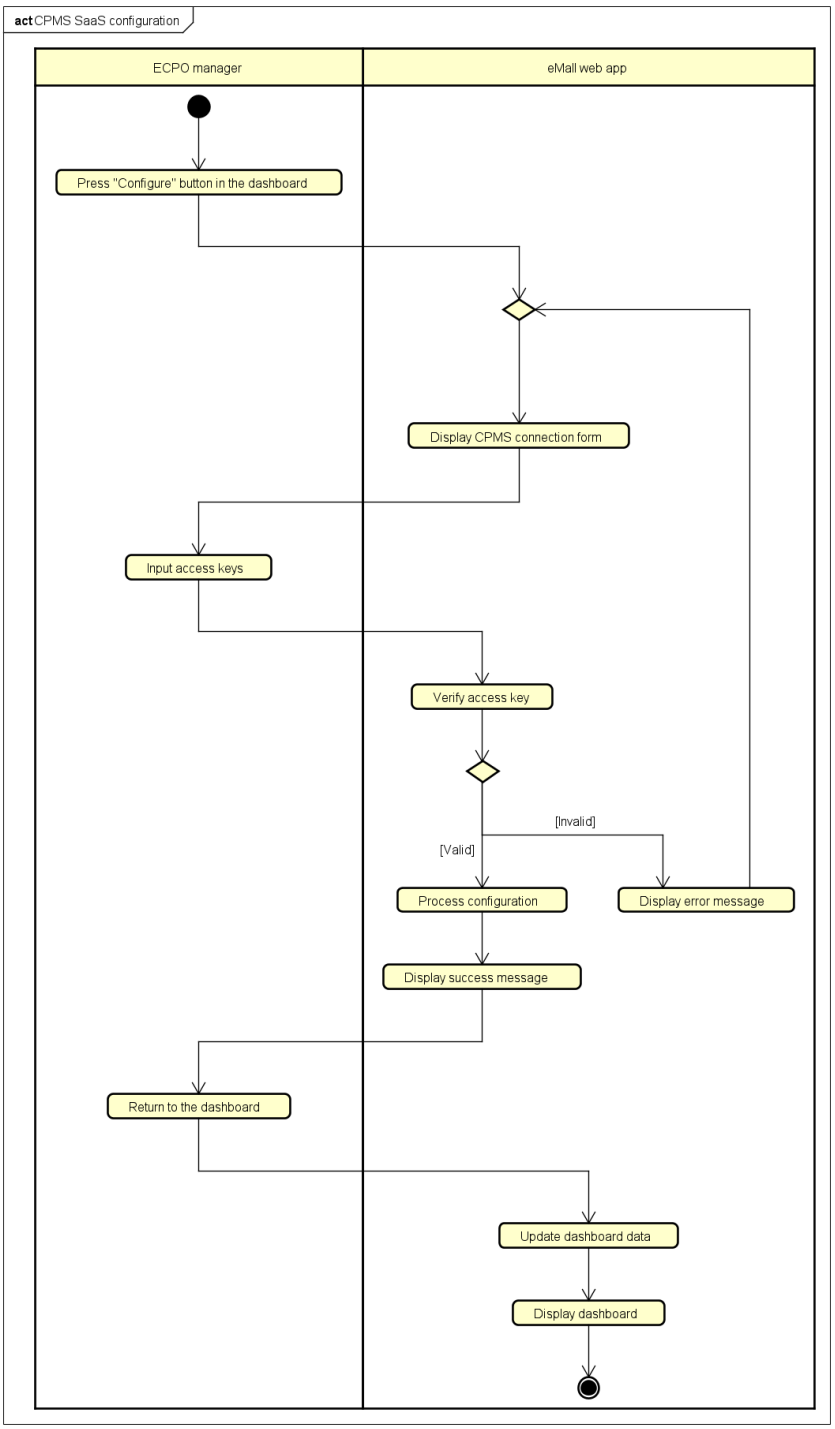

Figure 3.10: CPMS SaaS configuration activity diagram.

Client CPO manager login

Actor: client CPO manager.

Entry conditions

- The manager is an employee of the client CPO.
- The manager is on the login view of the client CPO's CPMS web interface.

Event flow

1. The manager clicks the "Sign in" button.
2. The CPMS displays a form where the manager has to input the login credentials provided by the client CPO.
3. The manager fills in the form and clicks the "Sign in" button.
4. The CPMS validates the data, creating a user session on the app, and displays the dashboard

Exceptions

- *The manager fails to input some of the required information*:
  the CPMS displays the same view, highlighting the missing fields and specifying that they have to be filled.
- *The manager inputs incorrect credentials*:
  the CPMS displays the same view with an error alert signaling the problem.

Browsing DSO energy prices

Actor: client CPO manager.

Entry conditions

- The manager is an employee of the client CPO.
- The manager has logged into the client CPO's CPMS web interface and is on the dashboard.

Event flow

1. The manager navigates to the "Energy sources" tab.
2. The CPMS displays the energy sources tab, which includes a summary of the active provision contracts and a list of available DSOs with their prices and recent trends.
3. The manager clicks on the "Filter" button above the DSO list.
4. The CPMS displays an overlay with the filtering options.
5. The manager selects the desired filters.

6. The CPMS displays the filtered results.

Exceptions

- *The CPMS cannot connect to one or more DSOs*:
  the CPMS displays an error alert signaling the missing connection, then displays the list without the unreachable DSOs.
- *The filtering conditions selected by the manager are invalid*:
  the CPMS displays an empty list and an alert warning about the invalid conditions, prompting the manager to change them.

Activity diagram: Figure 3.11.

Stipulating enegy provision contract

Actor: client CPO manager.

Entry conditions

- The manager is an employee of the client CPO.
- The manager has logged into the client CPO's CPMS web interface and is on the energy sources tab.

Event flow

1. The manager clicks on the "Manual selection" button.
2. The CPMS displays a form asking the manager to confirm their credentials before continuing.
3. The manager enters their password and clicks "Confirm".
4. The CPMS displays a modified energy sources tab, where each of the potential new sources is displayed next to a "Buy" button.
5. The manager clicks the "Buy" button next to the desired energy source.
6. The CPMS displays a form where the manager can specify the quantity of energy to buy as part of the contract.
7. The manager selects the quantity of energy to buy, then clicks the "Confirm" button.
8. The CPMS displays a success message and the manual selection energy sources tab.

Exceptions

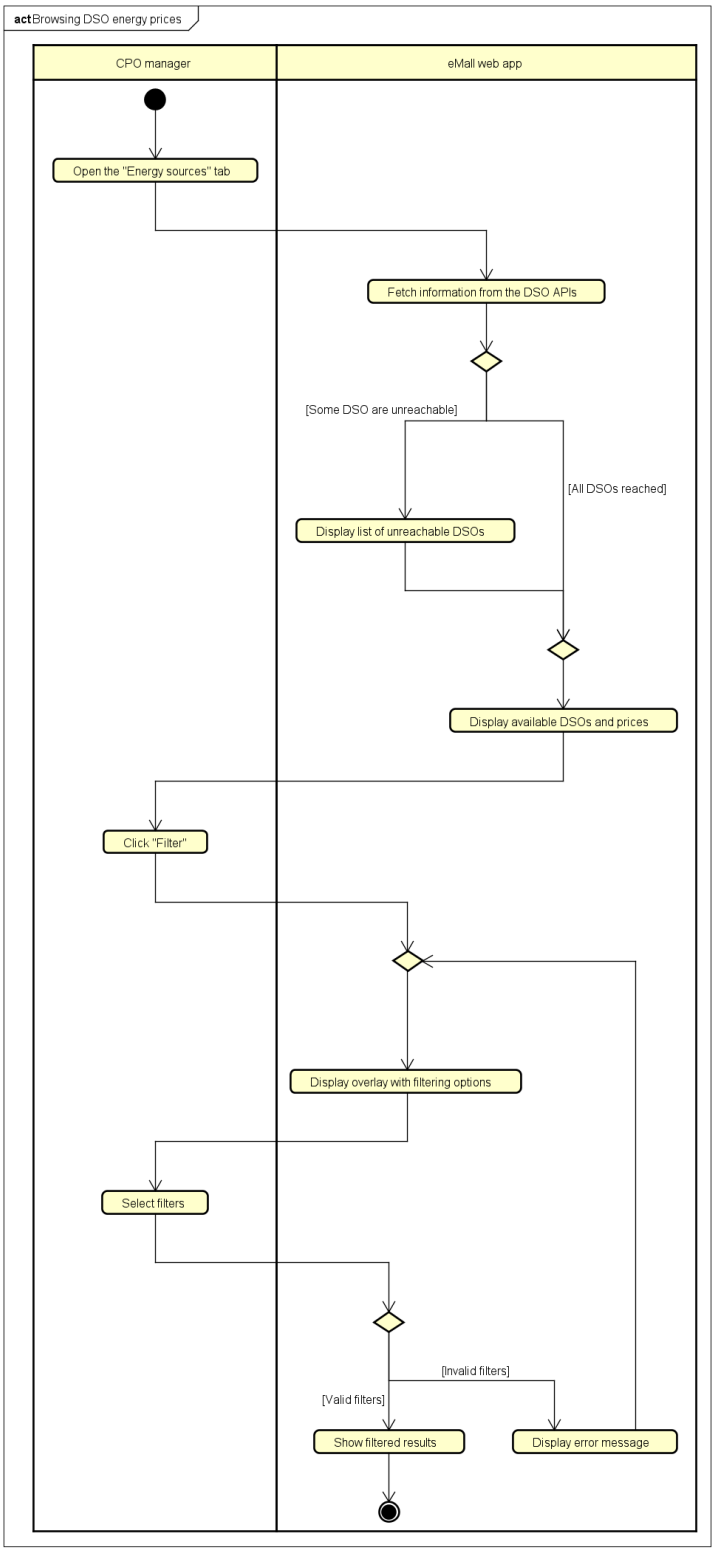

Figure 3.11: Browsing DSO energy prices activity diagram.

- *The manager inputs incorrect credentials when prompted*:
  the CPMS displays the same view with an error alert signaling the problem.

- *The CPMS cannot connect to one or more DSOs*:
  the CPMS displays an error alert signaling the missing connection, then displays the list without the unreachable DSOs.
- *The selected DSO cannot provide as much energy as requested*:
  the CPMS displays an error message specifying the maximum amount of energy the DSO can provide and prompts the manager to select again.

Activity diagram: Figure 3.12.

Selecting energy mix for a CP

Actor: client CPO manager.

Entry conditions

- The manager is an employee of the client CPO.
- The manager has logged into the client CPO's CPMS web interface and is on the dashboard.

Event flow

1. The manager navigates to the "Energy mix" tab.
2. The CPMS displays the energy mix tab, which includes a list of all the client CPO's CPs with the energy source currently being used and a "Manual selection" button.
3. The manager clicks on the "Manual selection" button next to a CP.
4. The CPMS displays a form asking the manager to confirm their credentials before continuing.
5. The manager enters their password and clicks "Confirm".
6. The CPMS diplays an energy mix pipeline view, where the manager can input sequential instructions (stages) about which energy source to use (DSO provision or on-site battery) and a condition which, when met, advances the pipeline to the following stage (when the pipeline advances past the last manually selected stage, the CP goes back to automatic selection).
7. The manager adds a first stage of battery energy to be erogated until the battery level goes under the 100 MWh threshold, and a second stage of DSO provision to be erogated until the end of the day, then clicks on the "Save" button.

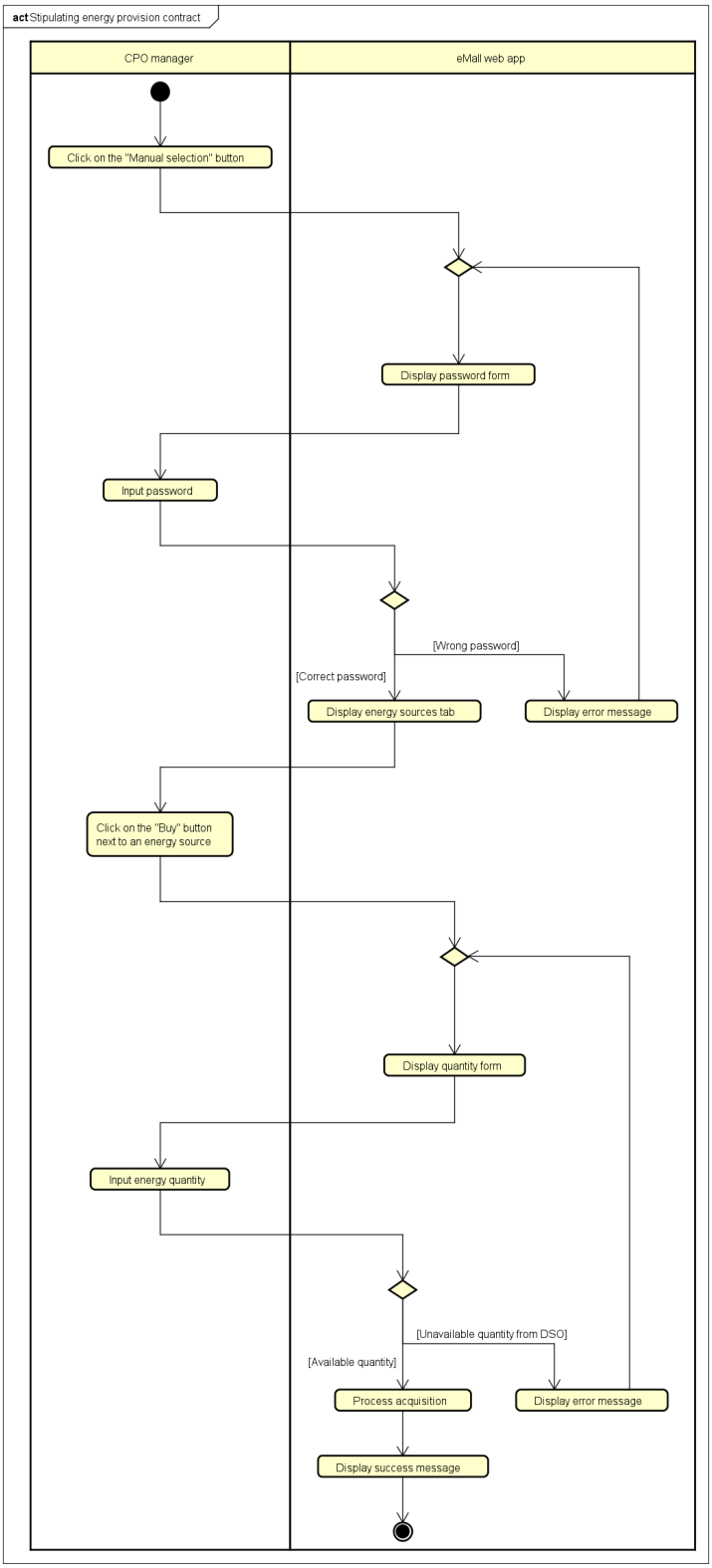

Figure 3.12: Stipulating energy provision contract activity diagram.

8. The CPMS displays the energy mix tab with the updated selection for the CP in question.

Exceptions

- *The manager inputs incorrect credentials when prompted*:
  the CPMS displays the same view with an error alert signaling the problem.
- *An unexpected error occurs that prevents the specified stages and conditions from being fulfilled*:
  the CPMS reverts to automatic selection.

Activity diagram: Figure 3.13.

3.3 Performance requirements

The application should provide real time information about the availability and usage of the charging sockets, with a response time ideally under 2 seconds for the EV users and even lower for the CPOs.

Note: response time of the eMSP can depend also on the CPMSs of external CPOs.

The management of energy offers from the DSOs is less time sensitive.

The amount of end-users could be in the order of tens or even hundreds of thousands, so the system should be able to withstand an appropriate amount of requests, considering also which are the busiest hours of the day.

It's important that the CPMS is always working (except for when the station is closed) in order to manage the charging process, also for safety reasons. The eMSP should also have the lowest downtime possible to provide good utility to the users.

3.4 Design constraints

3.4.1 Standards compliance

The CPMS must regulate the power provided by the charging sockets according to the IEC 61851 standard.

The CPMS also has to provide a REST API as described in the Communications Interfaces section. The API uses a key-based authentication standard.

Regarding privacy and security the system must comply with the General Data Protection

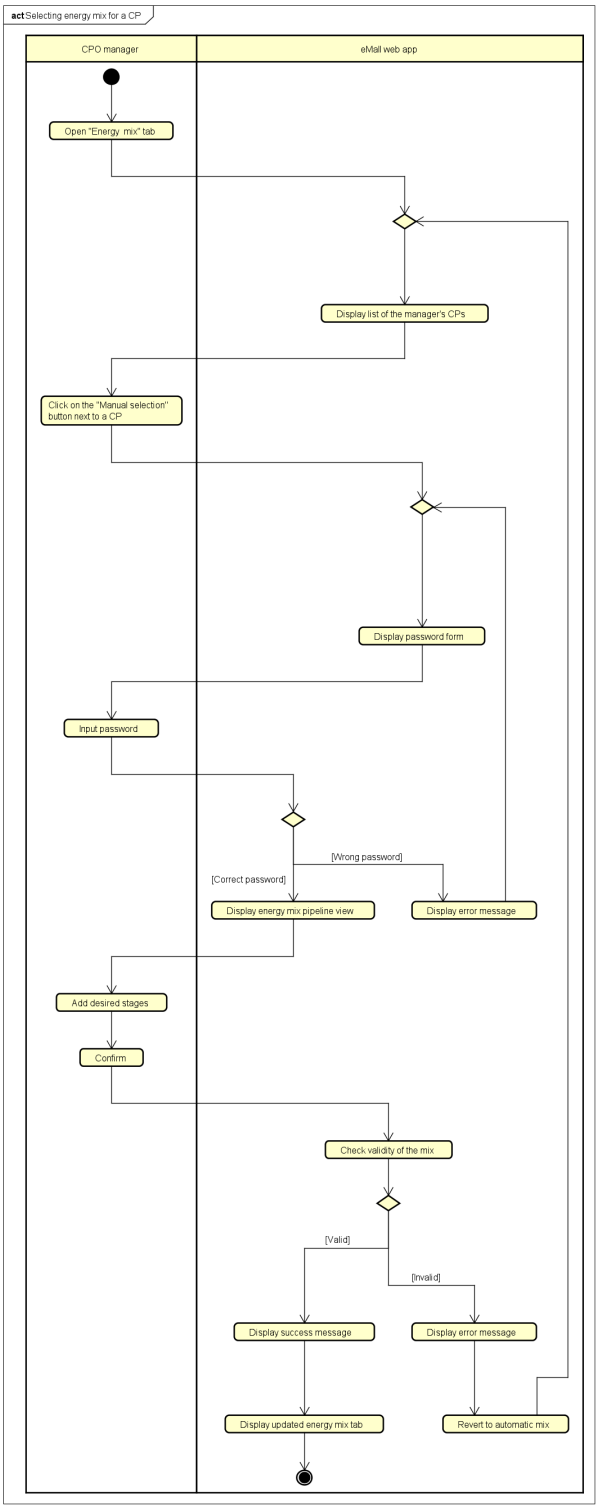

Figure 3.13: Selecting energy mix for a CP activity diagram

Regulation (GDPR) and use an external secure service to handle payments.

3.4.2 Hardware limitations

To use the mobile app, a device with an Android or iOS operating system is required. The device must have internet access and a camera to be able to scan QR codes.

For the web app, a device with a modern browser and internet access is required.

The charging sockets must have a display to show QR codes and charging information and must all be connected with the CPMS.

3.5 Software system attributes

3.5.1 Reliability

As mentioned before, it's important that the CPMS is very reliable because any fault reguarding the energy flow may represent a safety concern. There shouldn't be unexpected behaviors, all possible faults must be handled in a safe way.

The system overall should also be reliable for a good user experience, frequent crashes and errors might lead the users not to use the service anymore.

3.5.2 Availability

A good availability for the overall system would be 95

The downtime of the CPMS in particular should always be due to planned maintenance and preferably during the less busy periods of the day/year.

3.5.3 Security

Sensitive data about the users and their vehicles should be protected and encrypted when stored.

All communication containing private data (about the both the users and the CPOs) is secured with the TLS protocol.

The payments are handled by an external, secure authority.

Access to some funcitionalities of the CPMS such as the choice of an energy provider are allowed to authorized personnel only.

The API provided by the CPMS is not public since it can be used to retrieve information and book charging sockets, authentication is secured thanks to API keys.

3.5.4 Maintainability

Mantainability and scalability of the application are really important since eMobility is a fast evolving area, there might be the need to comply with new standard or regulations, to add new features and to keep up with the increasing number of users expected in the future.

3.5.5 Portability

The mobile application should be available to most mobile operating systems (i.e. Android and iOS) while the web app available to charging point operators should be accessible with any modern browser.

4 | Formal analysis using Alloy

The model signatures describe the objects of the analysis.

- `Actor` instances represent a stakeholder and can be of type `User` or `Company`. Users own EVs, while companies are further distinguished between CPOs and DSOs, who possess CSs and energy prices on the market, respectively. Finally, `CPO` instances are peculiarized by an energy stock.
- EVs and CSs are assigned standalone signatures, and they both involve an attribute of type `BatteryLevel`, which is nothing but an enumeration (i.e., a finite set of possible values) to capture the current state of the entity's batteries.
- `Socket` instances are part of `ChargingStation` ones. They are characterized by an element of the `SocketType` enumeration, the cost to be payed for a charge and at most a connected EV.
- Reservations include a timeframe and a user to properly enclose the act of booking a CSS.
- `BankAccount` and `Identifier` are supportive signatures to help conducting static and dynamic analyses on the model itself.

Figure 4.1 shows the overall structure of the Alloy model.

Facts are specifications which must hold in all the worlds compliant with the meta-model definitions and relationships. Apart from the straighforward ones, it should be underlined that a series of restrictions were further included in the generation of a plausible world (see Figure 4.2). This refers to the following:

- the `BankAccount` objects multiplicity in `Actor`'s specializations;
- the strengthening of `CPO_ChargingStation_Relationship` and `Vehicle_Id_Relationship` facts;
- a constraint to describe the uniqueness of car license plates.

Moreover, the weaker version of the listed facts are necessary to lead to sound and correct

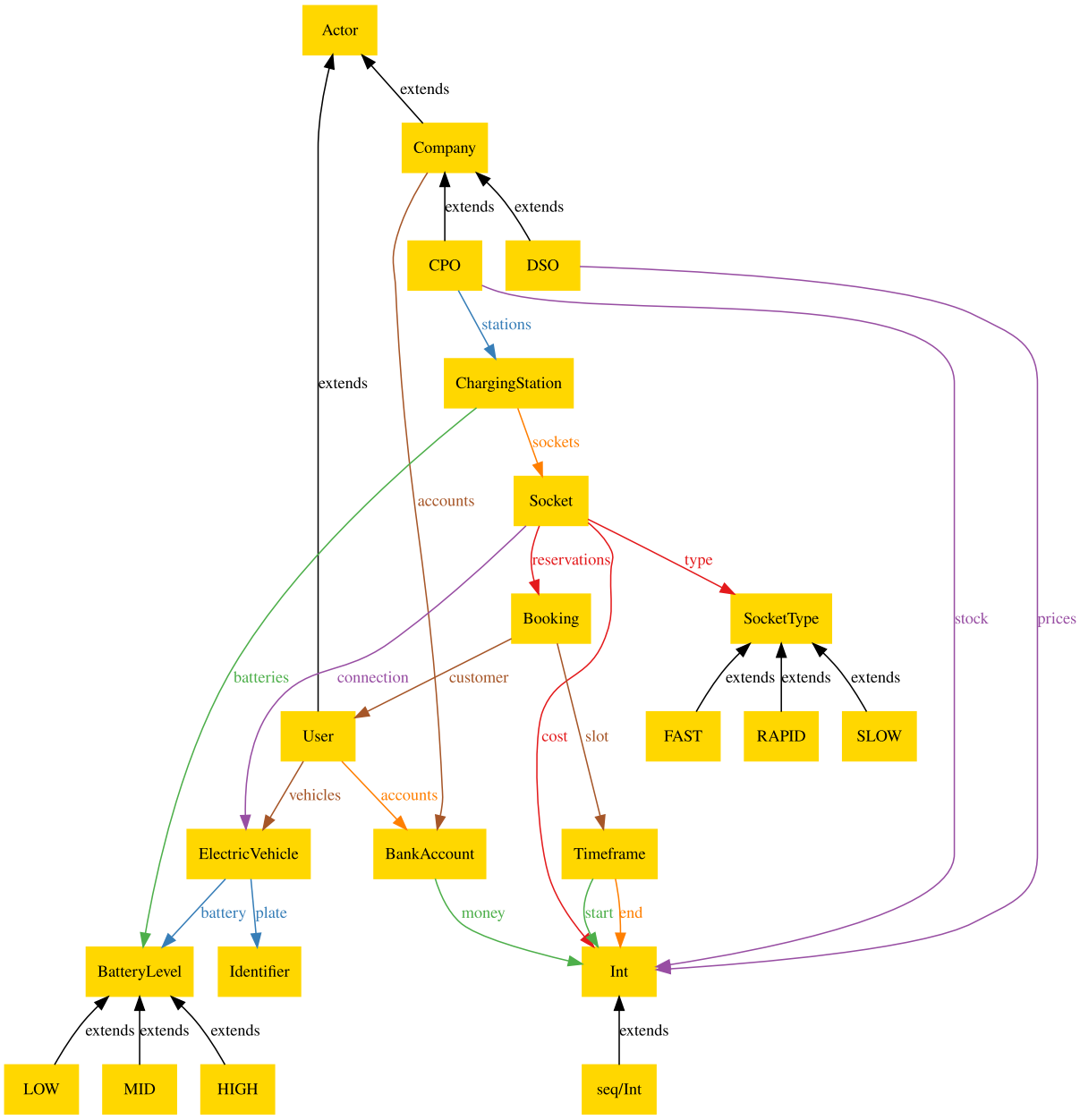

Figure 4.1: The meta-model generated by Alloy.

results in the dynamic analysis context (e.g., without allowing that two distinct instances of `ElectricVehicle` share the same plate number, the `User_CPO_Interaction` predicate would be inconsistent).

To conclude, there are a few of notable observations to be made.

- The car license plate, which is an object of type `Identifier`, introduces a functional dependency on `ElectricVehicle` instances. The world interpretation needs be shaped as follows: if two EVs share the same plate number, then the two EVs are

the same vehicle.

- The function `OverlappingBookings` focuses its attention on `Booking` instances, because a single `Timeframe` could be part of (i.e., reused by) multiple bookings.
- The `PresentTimeWorldConsistency` translates to *«If there exists a reservation which starts at the present time 0, then there must be a connected vehicle whose owner is equal to the user who booked the socket.»*.

```
// SIGNATURES
abstract sig Actor {}

sig User extends Actor {
    vehicles:   some ElectricVehicle,
    accounts:   some BankAccount
}

abstract sig Company extends Actor {
    accounts:   some BankAccount
}

sig CPO extends Company {
    stations:   some ChargingStation,
    stock:      one Int
}{
    stock > 0
}

sig DSO extends Company {
    prices: some Int
}{
    all p: Int | p in prices implies p > 0
}

sig ElectricVehicle {
    plate:      one Identifier,
    battery:    one BatteryLevel
}

sig ChargingStation {
    sockets:    some Socket,
    batteries:  set BatteryLevel
}
```

```
abstract sig BatteryLevel {}
one sig LOW, MID, HIGH extends BatteryLevel {} // L: 0%-33%, M: 34%-66%, H: 67%-100%

sig Socket {
    type:           one SocketType,
    cost:           one Int,
    connection:     lone ElectricVehicle,
    reservations:   set Booking
}{
    cost > 0
}

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

sig Timeframe {
    start:  one Int,
    end:    one Int
}

sig Booking {
    customer:   one User,
    slot:       one Timeframe
}

sig BankAccount {
    money:  one Int
}

sig Identifier {}

// FACTS
fact User_Vehicle_Relationship {
    all v: ElectricVehicle | (one u: User | v in u.vehicles)
}

fact Actor_Bank_Relationship {
    all a: BankAccount | ((one u: User | a in u.accounts) && (no c: Company | a in
↪   c.accounts)) || ((no u: User | a in u.accounts) && (one c: Company | a in
↪   c.accounts)) // Logical XOR.
}

fact CPO_ChargingStation_Relationship {
    all c: ChargingStation | #{cpo: CPO | c in cpo.stations} >= 1
```

```
}

fact CS_CSS_Relationship {
    all css: Socket | (one cs: ChargingStation | css in cs.sockets)
}

fact Vehicle_Id_Relationship {
    all id: Identifier | #{v: ElectricVehicle | v.plate = id} >= 1
}

fact Socket_Booking_Relationship {
    all b: Booking | #{s: Socket | b in s.reservations} >= 1
}

fun OverlappingBookings (s: Socket, t: Booking) : set Booking {
    {b: Booking | b != t && b in s.reservations && (b.slot.start <= t.slot.end &&
↪   b.slot.end >= t.slot.start)}  // Set comprehension.
}

fact NoOverlappings {
    all s: Socket, b: Booking | b in s.reservations implies #OverlappingBookings[s, b]
↪   = 0
}

fact PresentTimeWorldConsistency {
    all s: Socket, b: Booking | (b in s.reservations && b.slot.start = 0) implies (one
↪   v: ElectricVehicle, u: User | s.connection = v && b.customer = u && v in
↪   u.vehicles)
}

fact NoUbiquity {
    all v: ElectricVehicle | no disj s1, s2: Socket | v in s1.connection && v in
↪   s2.connection
}

fact Timeframe_Booking_Relationship {
    all t: Timeframe | #{b: Booking | t = b.slot} >= 1
}

fact ValidTimeframe {
    all t: Timeframe | t.start >= 0 && t.start < t.end
}

fact NoBankruptcy {
```

```
    no a: BankAccount | a.money < 0
}

// WORLDS
pred World {
    #User = 3
    #CPO >= 2
    #DSO = 1
    #ChargingStation >= 3
    #Socket >= 3
    #Booking >= 3
    all u: User | #u.accounts = 1 && all c: Company | #c.accounts = 1   // CONSTRAINT:
↪   BankAccount multiplicity
    all c: ChargingStation | (one cpo: CPO | c in cpo.stations)         // CONSTRAINT:
↪   CPO_ChargingStation_Relationship
    all id: Identifier | (one v: ElectricVehicle | v.plate = id)        // CONSTRAINT:
↪   Vehicle_Id_Relationship
    no disj v1, v2: ElectricVehicle | v1.plate = v2.plate               // CONSTRAINT:
↪   ElectricVehicle car licence plates
}
run World for 8

// DYNAMIC MODELING
pred Socket_Unchanged (s, s': Socket) {
    s.type = s'.type && s.cost = s'.cost && (one c: ChargingStation | s in c.sockets
↪   && s' in c.sockets)
}

pred User_Socket_Interaction_C (u: User, t: Timeframe, b: Booking, s, s': Socket) {
    /* The user reserves a socket, providing a timeframe. The bookings of the socket
↪   are updated. */
    b.customer = u && b.slot = t && b not in s.reservations && s'.reservations =
↪   s.reservations + b && Socket_Unchanged[s, s']
}

pred User_Socket_Interaction_D (u: User, t: Timeframe, b: Booking, s, s': Socket) {
    /* The user cancels a reservation. The bookings of the socket are updated. */
    b.customer = u && b.slot = t && b in s.reservations && s'.reservations =
↪   s.reservations - b && Socket_Unchanged[s, s']
}

pred User_Unchanged (u, u': User, v, v': ElectricVehicle, U_Bank, U_Bank':
↪   BankAccount) {
    u.vehicles - v = u'.vehicles - v' && u.accounts - U_Bank = u'.accounts - U_Bank'
```

```
}

pred CPO_Unchanged (cpo, cpo': CPO, CPO_Bank, CPO_Bank': BankAccount) {
    cpo.stations = cpo'.stations && cpo.accounts - CPO_Bank = cpo'.accounts -
↪   CPO_Bank'
}

pred Charge (v, v': ElectricVehicle) {
    (v.battery = LOW implies (v'.battery = MID || v'.battery = HIGH)) && ((v.battery =
↪   MID || v.battery = HIGH) implies v'.battery = HIGH)
}

pred Disconnection (v, v': ElectricVehicle, s, s': Socket) {
    (v in s.connection && v not in s'.connection) && (v' not in s.connection && v' not
↪   in s'.connection)
}

pred MoneyTransfer_User_CPO (u, u': User, cpo, cpo': CPO, U_Bank, U_Bank', CPO_Bank,
↪   CPO_Bank': BankAccount, cost: Int) {
    U_Bank in u.accounts && U_Bank' in u'.accounts && CPO_Bank in cpo.accounts &&
↪   CPO_Bank' in cpo'.accounts && minus[U_Bank.money, cost] >= 0 && U_Bank'.money =
↪   minus[U_Bank.money, cost] && CPO_Bank'.money = plus[CPO_Bank.money, cost]
}

pred User_CPO_Interaction (v, v': ElectricVehicle, u, u': User, s, s': Socket, cpo,
↪   cpo': CPO, U_Bank, U_Bank', CPO_Bank, CPO_Bank': BankAccount) {
    /* After a charge, the user pays for the service. */
    v in u.vehicles && v' in u'.vehicles && v != v' && v.plate = v'.plate && Charge[v,
↪   v'] && Disconnection[v, v', s, s'] && MoneyTransfer_User_CPO[u, u', cpo, cpo',
↪   U_Bank, U_Bank', CPO_Bank, CPO_Bank', s.cost] && User_Unchanged[u, u', v, v',
↪   U_Bank, U_Bank'] && Socket_Unchanged[s, s'] && (CPO_Unchanged[cpo, cpo', CPO_Bank,
↪   CPO_Bank'] && cpo.stock = cpo'.stock)
}

pred DSO_Unchanged (dso, dso': DSO, DSO_Bank, DSO_Bank': BankAccount) {
    dso.prices = dso'.prices && dso.accounts - DSO_Bank = dso'.accounts - DSO_Bank'
}

pred EnergyTransfer_CPO_DSO (cpo, cpo': CPO, energy: Int) {
    energy > 0 && cpo'.stock = plus[cpo.stock, energy]
}

pred MoneyTransfer_CPO_DSO (cpo, cpo': CPO, dso, dso': DSO, CPO_Bank, CPO_Bank',
↪   DSO_Bank, DSO_Bank': BankAccount, price: Int) {
```

```
    price in dso.prices && CPO_Bank in cpo.accounts && CPO_Bank' in cpo'.accounts &&
↪   DSO_Bank in dso.accounts && DSO_Bank' in dso'.accounts && minus[CPO_Bank.money,
↪   price] >= 0 && CPO_Bank'.money = minus[CPO_Bank.money, price] && DSO_Bank'.money =
↪   plus[DSO_Bank.money, price]
}

pred CPO_DSO_Interaction (cpo, cpo': CPO, dso, dso': DSO, e, p: Int, CPO_Bank,
↪   CPO_Bank', DSO_Bank, DSO_Bank': BankAccount) {
    /* The CPO buys energy from a DSO. */
    EnergyTransfer_CPO_DSO[cpo, cpo', e] && MoneyTransfer_CPO_DSO[cpo, cpo', dso,
↪   dso', CPO_Bank, CPO_Bank', DSO_Bank, DSO_Bank', p] && CPO_Unchanged[cpo, cpo',
↪   CPO_Bank, CPO_Bank'] && DSO_Unchanged[dso, dso', DSO_Bank, DSO_Bank']
}

// DYNAMIC ANALYSIS
pred Dyn_User_Socket_Interaction_C (u: User, t: Timeframe, b: Booking, s, s': Socket)
↪   {
    User_Socket_Interaction_C[u, t, b, s, s']
    all v: ElectricVehicle | (one u: User | v in u.vehicles)
    no disj v1, v2: ElectricVehicle | v1.plate = v2.plate
}
run Dyn_User_Socket_Interaction_C

pred Dyn_User_Socket_Interaction_D (u: User, t: Timeframe, b: Booking, s, s': Socket)
↪   {
    User_Socket_Interaction_D[u, t, b, s, s']
    all v: ElectricVehicle | (one u: User | v in u.vehicles)
    no disj v1, v2: ElectricVehicle | v1.plate = v2.plate
}
run Dyn_User_Socket_Interaction_D

assert CD_Ops_Interaction {
    all s, s', s'': Socket, u: User, t: Timeframe, b: Booking |
↪   User_Socket_Interaction_C[u, t, b, s, s'] && User_Socket_Interaction_D[u, t, b,
↪   s', s''] implies s.reservations = s''.reservations
}
check CD_Ops_Interaction

run User_CPO_Interaction for 8

run CPO_DSO_Interaction for 8
```

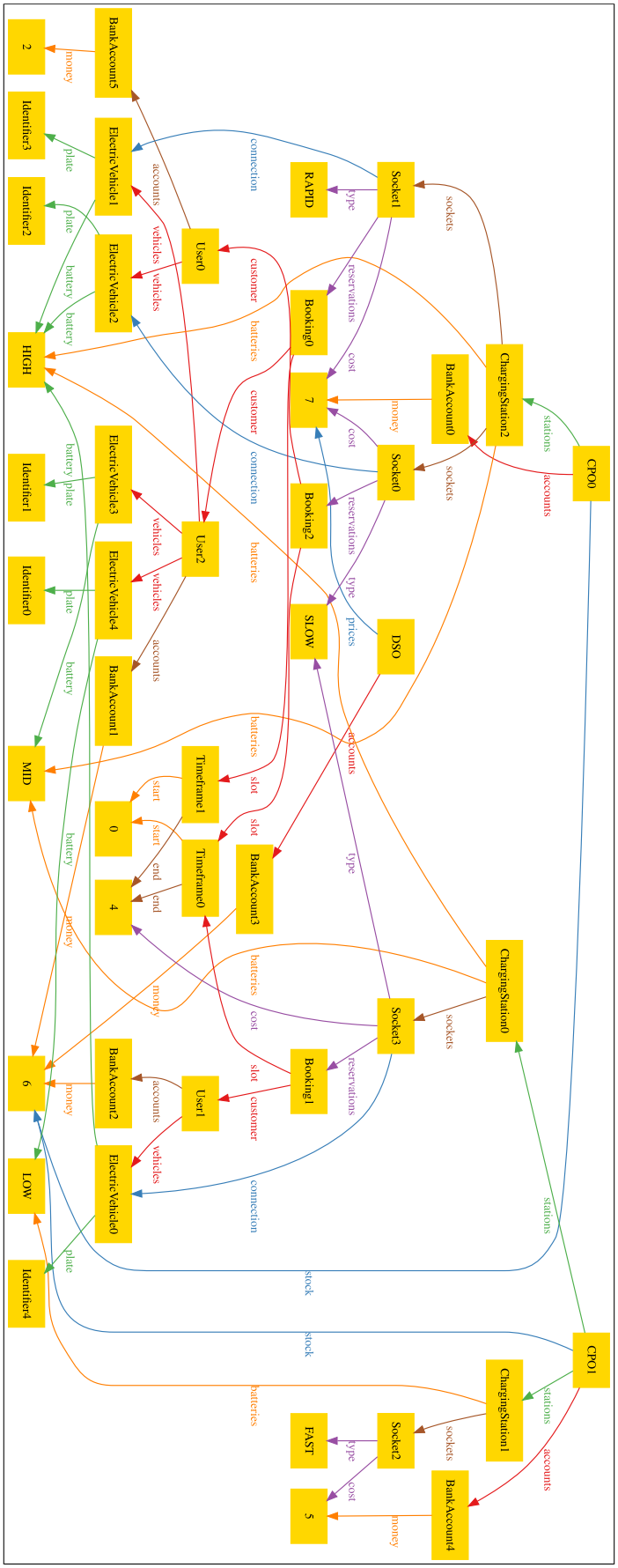

Figure 4.2: A sample world for the static analysis.

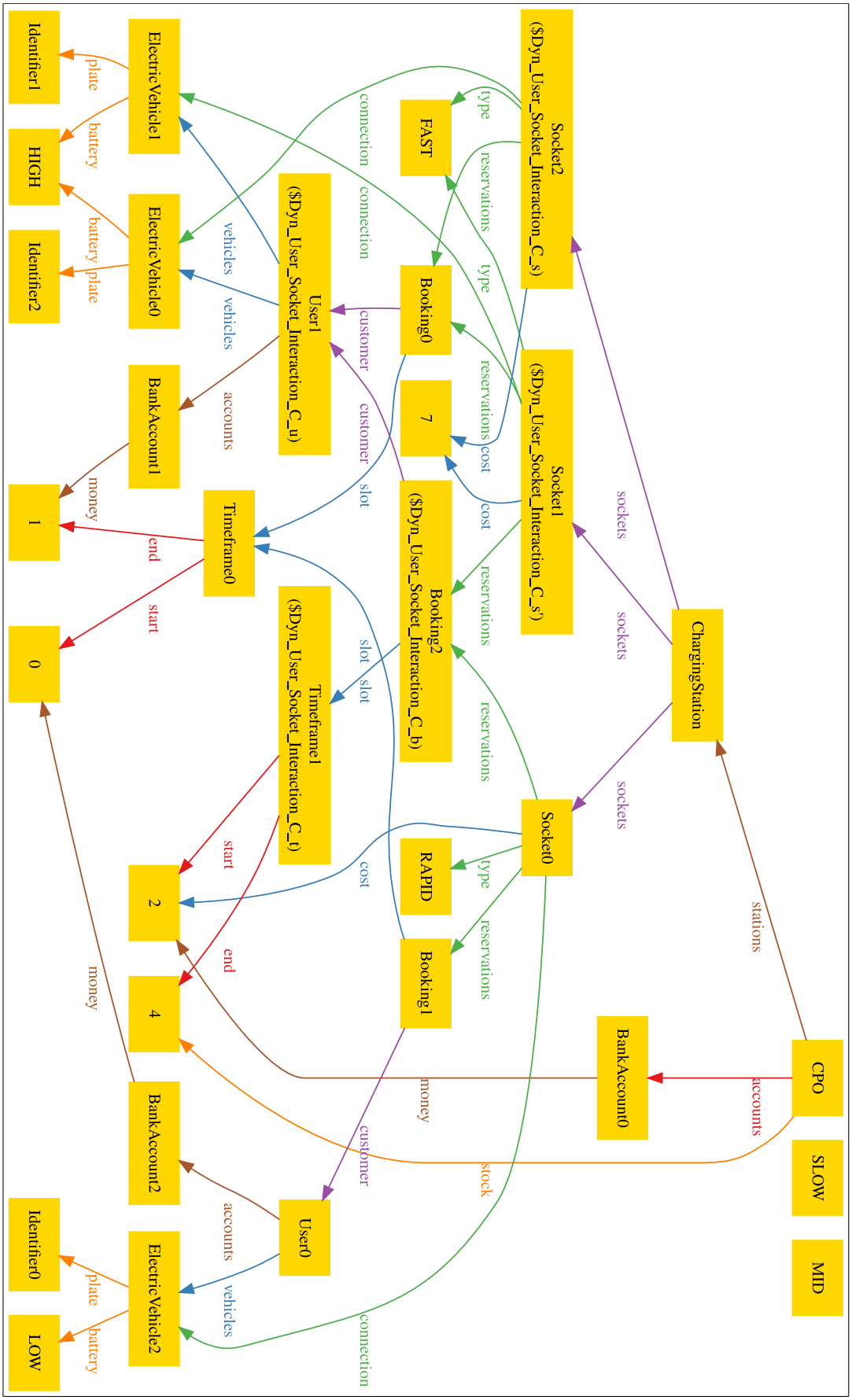

Figure 4.3: A world exemplifying the `User_Socket_Interaction_C` dynamic analysis predicate.

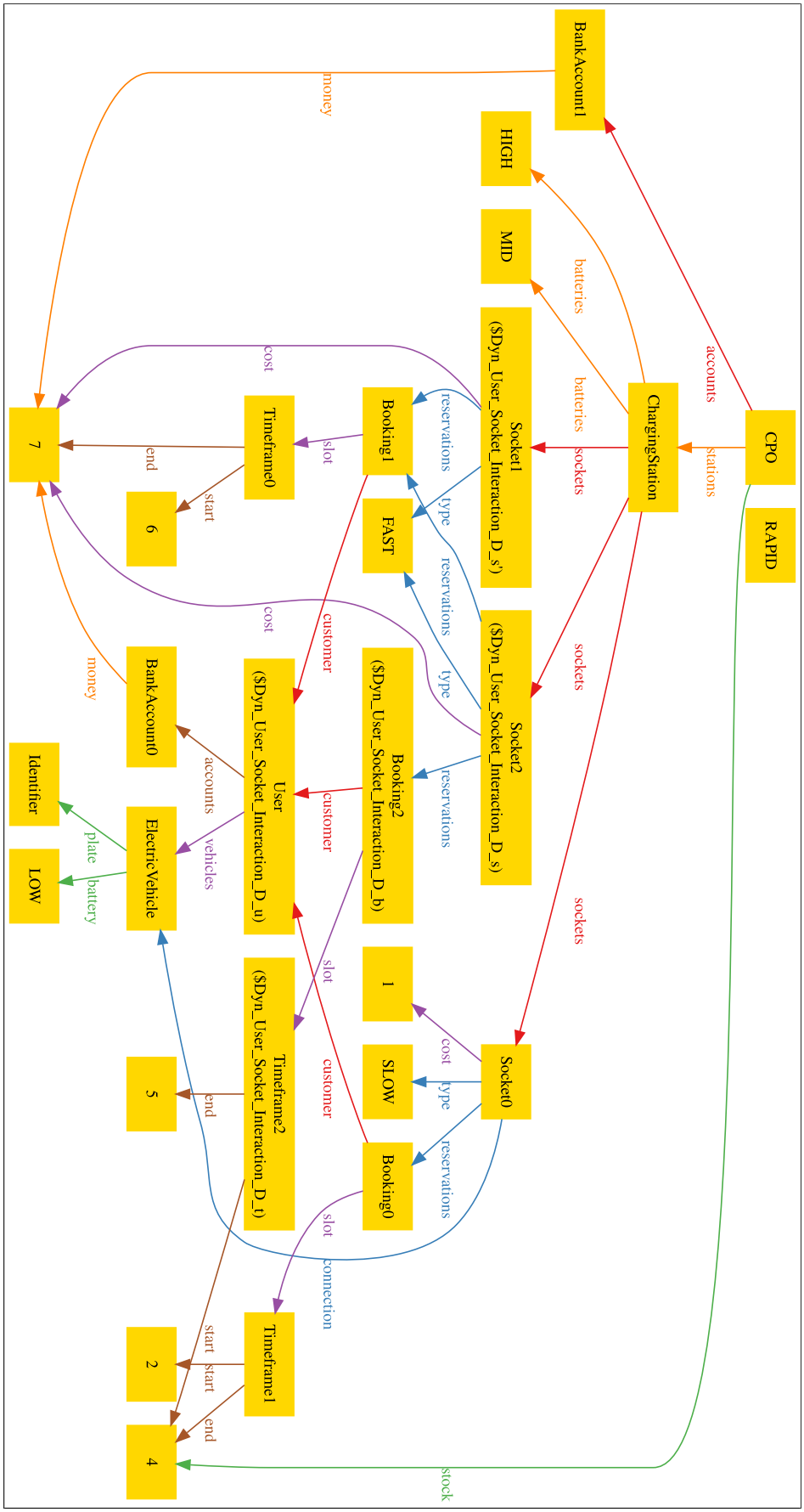

Figure 4.4: A world exemplifying the `User_Socket_Interaction_D` dynamic analysis predicate.

Figure 4.5: A world exemplifying the `User_CPO_Interaction` dynamic analysis predicate.

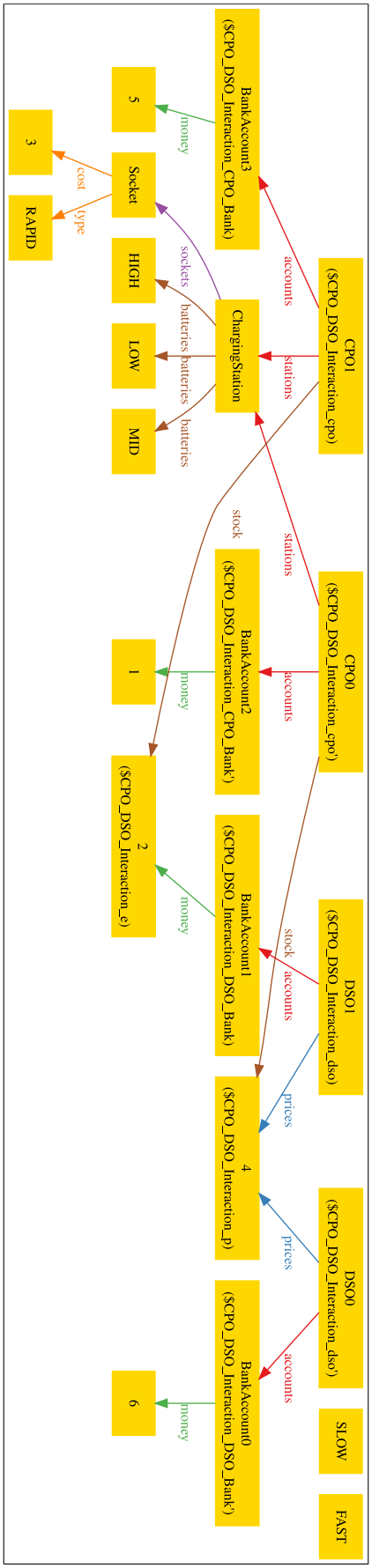

Figure 4.6: A world exemplifying the `CPO_DSO_Interaction` dynamic analysis predicate.

5 | Temporal workload summary

Chapter	Team member	Summary
Introduction	Tommaso Bonetti	12 hrs.
	Fabio Ciani	16 hrs.
	Davide Mozzi	10 hrs.
Overall description	Tommaso Bonetti	18 hrs.
	Fabio Ciani	19 hrs.
	Davide Mozzi	16 hrs.
Specific requirements	Tommaso Bonetti	30 hrs.
	Fabio Ciani	1 hrs.
	Davide Mozzi	22 hrs.
Formal analysis using Alloy	Tommaso Bonetti	0.5 hrs.
	Fabio Ciani	22 hrs.
	Davide Mozzi	0.5 hrs.

6 | References

[1] Jackson, D. *Software Abstractions: Logic, Language, and Analysis*. MIT Press, 2016.